Anhang A.1 – BaMainProgram

```
# -*- coding: utf-8 -*-
"""
Created on Mon May 12 09:01:11 2014

@title:   AMODD - Analysing Module of Dose Distributions
@author:  Felix Voigt
@version: 1.0
@date:    18.07.2014
"""

import sys

from PyQt4 import QtGui
from BaQtDesigner import Ui_MainWindow
from numpy import *
from PIL import Image, ImageDraw
#from reportlab.platypus import Paragraph, SimpleDocTemplate, Spacer
#from reportlab.lib.styles import getSampleStyleSheet
from reportlab.pdfgen.canvas import Canvas
import os.path
import time


# Constants for the calculations of the scan-values of the Wedged Phantom
X_DDWP_P1=300+2*40    #  40mm right of the middl
X_DDWP_P4=300-2*40    #  40mm left of the middl

X_DDWP_P2_1=300+2*40    #  40mm right of the middl
X_DDWP_P2_2=300+2*60    #  60mm right of the middl
X_DDWP_P2_3=300+2*80    #  80mm right of the middl
X_DDWP_P3_1=300-2*80    #  80mm left of the middl
X_DDWP_P3_2=300-2*60    #  60mm left of the middl
X_DDWP_P3_3=300-2*40    #  40mm left of the middl

Y_DDWP_P1_1=Y_DDWP_P4_3=300-2*40    #  40mm above the middl
Y_DDWP_P1_2=Y_DDWP_P4_2=300-2*60    #  60mm above the middl
Y_DDWP_P1_3=Y_DDWP_P4_1=300-2*80    #  80mm above the middl

Y_DDWP_P2=Y_DDWP_P3=300+2*40    #  40mm below the middl



X_DDWP_P_OFFSET=2*8    # 8mm x direction
Y_DDWP_P_OFFSET=2*8    # 8mm y direction

# Constants for the calculations of the lateral dose of the Blue Phantom
COORD_DEVIATION=0.5

# General Constants
PASS=True
```

```
"""
  Klasse zur Beschreibung der Graphischen Benutzeroberfläche
"""
class MainWindowGui(QtGui.QMainWindow, Ui_MainWindow):
    """
      Initialisierung Graphischer Benutzeroberfläche
    """
    def __init__(self,  parent = None):
        # Laden des Konstructors von QWidget
        QtGui.QWidget.__init__(self, parent)
        # Laden der Qt-Designer Befehle
        self.setupUi(self)


##########################################################################
###########
#################################### Methods
#####################################

    def HideAll(self):                                          # ready
#•        hide all group boxes of all analyzing modules
#•        hide every widget/diagram
#•        clear all data in every widget/diagram
#•        hide every image

        gui.groupBox_DD_BluePhantom.setHidden(True)
        gui.groupBox_DD_WedgePhantom.setHidden(True)
        gui.groupBox_LateralDose_crossline.setHidden(True)
        gui.groupBox_LateralDose_inline.setHidden(True)

        gui.widget_0.canvas.axes.lines = []
        gui.widget_1.canvas.axes.lines = []
        gui.widget_2.canvas.axes.lines = []
        gui.widget_0.setHidden(True)
        gui.widget_1.setHidden(True)
        gui.widget_2.setHidden(True)

        gui.label_image.setHidden(True)
        gui.label_master_fail_pass.setHidden(True)

#-------------------------------------------------------------------------
    def FileToOpen(self):                                       # ready
#•        open file dialog to choose the input file
#•        show file path on the GUI
#
#       o return:
#            □        string of the file path

        print "FileToOpen"
        fileToOpen = QtGui.QFileDialog.getOpenFileName(gui, 'Open File')
        fileToOpen = str(fileToOpen.replace("/","\\"))
        gui.lineEdit_FilePath.setText(fileToOpen)
        print gui.lineEdit_FilePath.text()
        return (fileToOpen)
```

```
#--------------------------------------------------------------------------
    def LoadFromLYNXFile(self,inputf):                              # ready
#       o inputf: string of the file to load
#
#•        load datafrom Lynx-txt-file in array as float values
#             starting after 33 rows
#             (beginning of the data, first column is the y-coordinate)
#•        extract y-coordinates from the first column into a separate array
#•        create new data array without the first column(y-coordinate)
#•        load data in array as string after 32 rows
#             (first row is the x-coordinates)
#•        extract x-coordinates as float values from the first row
#             into a separate array
#
#       o return:
#            □       array of the dose-values
#            □       array of the x-coordinates
#            □       array of the y-coordinates
# LYNX-coordinates: bottom left: -150/-150 top right: 150/150
# file-coordinates: bottom left: -150/150 top right: 150/-150
#       so the values had to flip horizontal

        print "LoadFromLYNXFile"
        try:
            data_arr = loadtxt(inputf, delimiter = '\t', skiprows = 33)
            data_arr = data_arr[::-1,:]
            y_axis = data_arr[:,0]
            data_arr = data_arr[:,1:]
            x_arr = loadtxt(inputf, delimiter = '\t', skiprows = 32,
                        dtype = str)
            x_axis = array(x_arr[0, 1:], dtype = float)
        except IOError:
            print("Please correct your inputdata-path! ")

        return data_arr,x_axis,y_axis
#--------------------------------------------------------------------------
#Load Data from OPA-txt-file and save koordinates and doses in arrays
    def LoadDatafromOPA(self, inputfile,x):                         # ready
#       o inputfile: string of the file to load (text file)
#       o x: number of measurement series
#
#•        read the text file line wise and save the lines in a list
#•        search file line wise by table headings and endings and saver there line indices
#•        save x-, y-, z-coordinates and doses of the x. measurement series as float-
arrays
#
#       o return:
#            □       array of the x-coordinates
#            □       array of the y-coordinates
#            □       array of the z-coordinates
#            □       array of the dose-values

        print "LoadDDDatafromOPA"
        x=x-1
        if x<0: x=0
```

```
        try:
            liste= open(inputfile).readlines()
        except IOError:
            print("Please correct your inputdata-path! ")

        startlines=[]
        endlines=[]
        i=1
        k=0
        while i:
            try:
                k=liste.index("#\t  X      Y      Z     Dose\n",k,len(liste))
                startlines.append(k+2)
                k=liste.index(":EOM  # End of Measurement\n",k,len(liste))
                endlines.append(k)
            except ValueError:
                i=0
        try:
            len_measur=endlines[x]-startlines[x]
            gui.textEdit_MessageField.setText("Measurment series number " + \
                                    str(x+1) + " loaded.")
        except IndexError:
            gui.textEdit_MessageField.setText("There are not enought"+\
                                    " measurement series in the" +\
                                    " given data. \nYou selected"+\
                                    " the "+str(x+1)+\
                                    ". one but there ar only "+\
                                    str(len(startlines)))

            return array([0]), array([0]), array([0]), array([0])
        x_array=zeros(len_measur)
        y_array=zeros(len_measur)
        z_array=zeros(len_measur)
        d_array=zeros(len_measur)
        i=startlines[x]
        if float(liste[i][19:27])>float(liste[i+1][19:27]):
            while i<endlines[x]:                                    # reading the table from
bottom to top
                x_array[len(x_array)+startlines[x]-i-1]=float(liste[i][3:11])       # writing the data
from the last to the first index of the array
                y_array[len(y_array)+startlines[x]-i-1]=float(liste[i][11:19])
                z_array[len(z_array)+startlines[x]-i-1]=float(liste[i][19:27])
                d_array[len(d_array)+startlines[x]-i-1]=float(liste[i][27:35])
                i=i+1
        else:
            while i<endlines[x]:
                x_array[i-startlines[x]]=float(liste[i][3:11])
                y_array[i-startlines[x]]=float(liste[i][11:19])
                z_array[i-startlines[x]]=float(liste[i][19:27])
                d_array[i-startlines[x]]=float(liste[i][27:35])
                i=i+1

        return x_array, y_array, z_array, d_array
#---------------------------------------------------------------------------
    def P_Deviation(self,s1,s2):
```

```
#       o s1: string of values separated by “/” (e.g. “70/50/30”)
#       o s2: string of values separated by “/” (e.g. “60/50/45”)
#
#•        split strings in single Values
#•        calculate absolute difference of the values
#
#       o return:
#            □        string of the differences separated by “/” (e.g. “10/0/15”)

        print "P_Deciation",s1,s2
        if s1=="-":
            return "-"
        if s2=="-":
            return "-"
        a1=s1.split("/")
        a2=s2.split("/")
        t1=abs(float(a1[0])-float(a2[0]))
        t2=abs(float(a1[1])-float(a2[1]))
        t3=abs(float(a1[2])-float(a2[2]))
        return str(t1)+"/"+str(t2)+"/"+str(t3)
#-------------------------------------------------------------------------------
    def P_Deviation_Compare(self, s1, s2):
#       o s1: string of values separated by “/” (e.g. “10/0/15”)
#       o s2: string of a tolerance value (e.g. “10”)
#
#•        split string s1 in single Values
#•        check if every single value of s1 is less or equal s2 (as float values)
#
#       o return:
#            □        True or False

        print "P_Deviation_Compare",s1,s2
        if s1=="-":
            return True
        if s2=="-":
            return True
        a1=s1.split("/",2)
        for i in range(len(a1)):
            if float(a1[i])>float(s2):
                return False
        return True
#-------------------------------------------------------------------------------
    def LoadDQADSDDLYNXIV(self, option, refFile):                    # ready
#(“Load Daily Quality Assurance Double
#                    Scattering Depth Dose LYNX ‘Ideal’ Values”)
#       o option: String of the option, which has to be loaded
#
#•        load text of file “Depth Dose - Wedge Phantom - LYNX.txt” and
#                     split it by tabs
#•        show reference values, tolerance values und deviation between
#                     measured values and reference values on the GUI
#•        compare deviation values to tolerance values,
#                    color them red or lightgreen
#•        if option not loadable, all values(except the measured values)
#                    get the string “n.a.”
```

```
        gui.textEdit_MessageField.setText("")
        gui.lineEdit_DDWP_option.setText("")
        print "DSDDLYNXIV"
        liste =loadtxt(refFile, delimiter = '\t', skiprows=2, dtype=str)
        val=True
        for i in range(len(liste)):
            if option==liste[i][0]:
                gui.label_master_fail_pass.setHidden(False)
                PASS=True
                val=False
                gui.label_DDWP_RangeModulation.setText(liste[i][1] + \
                                            "  " + liste[i][2] +\
                                            "  ("+ option +")")
                gui.label_DDWP_R1_ival.setText(liste[i][3])
                gui.label_DDWP_R1_tol.setText(liste[i][4])
                gui.label_DDWP_R2_ival.setText(liste[i][5])
                gui.label_DDWP_R2_tol.setText(liste[i][6])
                gui.label_DDWP_P1_ival.setText(liste[i][7])
                gui.label_DDWP_P1_tol.setText(liste[i][8])
                gui.label_DDWP_P2_ival.setText(liste[i][9])
                gui.label_DDWP_P2_tol.setText(liste[i][10])
                gui.label_DDWP_P3_ival.setText(liste[i][11])
                gui.label_DDWP_P3_tol.setText(liste[i][12])
                gui.label_DDWP_P4_ival.setText(liste[i][13])
                gui.label_DDWP_P4_tol.setText(liste[i][14])

# Daily QA DD-values calculate deviation
                gui.label_DDWP_R1_dev.setText(str(abs(\
                        float(gui.label_DDWP_R1_rval.text())-\
                        float(gui.label_DDWP_R1_ival.text()))))
                gui.label_DDWP_R2_dev.setText(str(abs(\
                        float(gui.label_DDWP_R2_rval.text())-\
                        float(gui.label_DDWP_R2_ival.text()))))

                gui.label_DDWP_P1_dev.setText(gui.P_Deviation(\
                            gui.label_DDWP_P1_rval.text(),\
                            gui.label_DDWP_P1_ival.text()))
                gui.label_DDWP_P2_dev.setText(gui.P_Deviation(\
                            gui.label_DDWP_P2_rval.text(),\
                            gui.label_DDWP_P2_ival.text()))
                gui.label_DDWP_P3_dev.setText(gui.P_Deviation(\
                            gui.label_DDWP_P3_rval.text(),\
                            gui.label_DDWP_P3_ival.text()))
                gui.label_DDWP_P4_dev.setText(gui.P_Deviation(\
                            gui.label_DDWP_P4_rval.text(),\
                            gui.label_DDWP_P4_ival.text()))
# Daily QA DD-values compare deviation to tolerance
                if(float(gui.label_DDWP_R1_dev.text())<\
                            float(gui.label_DDWP_R1_tol.text())):
                    gui.label_DDWP_R1_dev.setStyleSheet(\
                            "QLabel {Background-color:lightgreen}")
                else:
                    PASS=False
                    gui.label_DDWP_R1_dev.setStyleSheet(\
```

```
                "QLabel {Background-color:red}")
    if(float(gui.label_DDWP_R2_dev.text())<\
                float(gui.label_DDWP_R2_tol.text())):
        gui.label_DDWP_R2_dev.setStyleSheet(\
                "QLabel {Background-color:lightgreen}")
    else:
        PASS=False
        gui.label_DDWP_R2_dev.setStyleSheet(\
                "QLabel {Background-color:red}")

    if(gui.P_Deviation_Compare(gui.label_DDWP_P1_dev.text(),\
                        gui.label_DDWP_P1_tol.text())):
        gui.label_DDWP_P1_dev.setStyleSheet(\
                "QLabel {Background-color:lightgreen}")
    else:
        PASS=False
        gui.label_DDWP_P1_dev.setStyleSheet(\
                "QLabel {Background-color:red}")
    if(gui.P_Deviation_Compare(gui.label_DDWP_P2_dev.text(),\
                        gui.label_DDWP_P2_tol.text())):
        gui.label_DDWP_P2_dev.setStyleSheet(\
                "QLabel {Background-color:lightgreen}")
    else:
        PASS=False
        gui.label_DDWP_P2_dev.setStyleSheet(\
                "QLabel {Background-color:red}")
    if(gui.P_Deviation_Compare(gui.label_DDWP_P3_dev.text(),\
                        gui.label_DDWP_P3_tol.text())):
        gui.label_DDWP_P3_dev.setStyleSheet(\
                "QLabel {Background-color:lightgreen}")
    else:
        PASS=False
        gui.label_DDWP_P3_dev.setStyleSheet(\
                "QLabel {Background-color:red}")
    if(gui.P_Deviation_Compare(gui.label_DDWP_P4_dev.text(),\
                        gui.label_DDWP_P4_tol.text())):
        gui.label_DDWP_P4_dev.setStyleSheet(\
                "QLabel {Background-color:lightgreen}")
    else:
        PASS=False
        gui.label_DDWP_P4_dev.setStyleSheet(\
                "QLabel {Background-color:red}")

    if PASS:
        gui.label_master_fail_pass.setText("PASS")
        gui.label_master_fail_pass.setStyleSheet(\
                    "QLabel {Background-color:green}")
    else:
        gui.label_master_fail_pass.setText("CAUTION!")
        gui.label_master_fail_pass.setStyleSheet(\
                    "QLabel {Background-color:orange}")
```

```
        if val:
            gui.label_DDWP_RangeModulation.setText(option+" n.a.")
            gui.label_DDWP_R1_ival.setText("n.a.")
            gui.label_DDWP_R1_tol.setText("n.a.")
            gui.label_DDWP_R1_dev.setText("n.a.")
            gui.label_DDWP_R2_ival.setText("n.a.")
            gui.label_DDWP_R2_tol.setText("n.a.")
            gui.label_DDWP_R2_dev.setText("n.a.")
            gui.label_DDWP_P1_ival.setText("n.a.")
            gui.label_DDWP_P1_tol.setText("n.a.")
            gui.label_DDWP_P1_dev.setText("n.a.")
            gui.label_DDWP_P2_ival.setText("n.a.")
            gui.label_DDWP_P2_tol.setText("n.a.")
            gui.label_DDWP_P2_dev.setText("n.a.")
            gui.label_DDWP_P3_ival.setText("n.a.")
            gui.label_DDWP_P3_tol.setText("n.a.")
            gui.label_DDWP_P3_dev.setText("n.a.")
            gui.label_DDWP_P4_ival.setText("n.a.")
            gui.label_DDWP_P4_tol.setText("n.a.")
            gui.label_DDWP_P4_dev.setText("n.a.")
            gui.textEdit_MessageField.setText("Requested option is "+\
                                    "not available")
            gui.label_DDWP_R1_dev.setStyleSheet(\
                                    "QLabel {Background-color:none}")
            gui.label_DDWP_R2_dev.setStyleSheet(\
                                    "QLabel {Background-color:none}")
            gui.label_DDWP_P1_dev.setStyleSheet(\
                                    "QLabel {Background-color:none}")
            gui.label_DDWP_P2_dev.setStyleSheet(\
                                    "QLabel {Background-color:none}")
            gui.label_DDWP_P3_dev.setStyleSheet(\
                                    "QLabel {Background-color:none}")
            gui.label_DDWP_P4_dev.setStyleSheet(\
                                    "QLabel {Background-color:none}")
#-------------------------------------------------------------------------------
    def LoadLDBPIV(self,option, refFile):                                   # ready
#("Load Lateral Dose Blue Phantom 'Ideal' Values")
#       o option: String of the option, which has to be loaded
#
#•      load text of file "Lateral Dose - Water Phantom.txt" and
#               split it by tabs
#•      show reference values, tolerance values und deviation between
#               measured values and reference values on the GUI
#•      compare deviation values to tolerance values,
#               color them red or lightgreen
#•      if option not loadable, all values(except the measured values)
#               get the string "n.a."

        gui.textEdit_MessageField.setText("")
        gui.lineEdit_LDBP_option.setText("")
        gui.label_LDBP_RangeModulation.setText(option)
        print"LoadLDBPIV"
        liste =loadtxt(refFile, delimiter = '\t', skiprows=3, dtype = str)
        val=True
        for i in range(len(liste)):
```

```
            if option==liste[i][0] or option==liste[i][1]:
                gui.label_master_fail_pass.setHidden(False)
                PASS=True
                val=False
                gui.label_LDBP_RangeModulation.setText(liste[i][1]+
                                        "("+liste[i][0]+")")
                gui.label_LDBP_IL_FieldSize_ival.setText(liste[i][2])
                gui.label_LDBP_IL_FieldSize_tol.setText(liste[i][3])
                gui.label_LDBP_IL_Pen_low_ival.setText(liste[i][4])
                gui.label_LDBP_IL_Pen_low_tol.setText(liste[i][5])
                gui.label_LDBP_IL_Pen_high_ival.setText(liste[i][6])
                gui.label_LDBP_IL_Pen_high_tol.setText(liste[i][7])
                gui.label_LDBP_IL_LatOff_ival.setText(liste[i][8])
                gui.label_LDBP_IL_LatOff_tol.setText(liste[i][9])
                gui.label_LDBP_IL_MinMax_ival.setText(liste[i][10])
                gui.label_LDBP_IL_MinMax_tol.setText(liste[i][11])
                gui.label_LDBP_CL_FieldSize_ival.setText(liste[i][12])
                gui.label_LDBP_CL_FieldSize_tol.setText(liste[i][13])
                gui.label_LDBP_CL_Pen_low_ival.setText(liste[i][14])
                gui.label_LDBP_CL_Pen_low_tol.setText(liste[i][15])
                gui.label_LDBP_CL_Pen_high_ival.setText(liste[i][16])
                gui.label_LDBP_CL_Pen_high_tol.setText(liste[i][17])
                gui.label_LDBP_CL_LatOff_ival.setText(liste[i][18])
                gui.label_LDBP_CL_LatOff_tol.setText(liste[i][19])
                gui.label_LDBP_CL_MinMax_ival.setText(liste[i][20])
                gui.label_LDBP_CL_MinMax_tol.setText(liste[i][21])

# LD-values calculate deviation
                gui.label_LDBP_IL_FieldSize_dev.setText(str(abs(\
                        float(gui.label_LDBP_IL_FieldSize_rval.text())-\
                        float(gui.label_LDBP_IL_FieldSize_ival.text()))))
                gui.label_LDBP_IL_Pen_low_dev.setText(str(abs(\
                        float(gui.label_LDBP_IL_Pen_low_rval.text())-\
                        float(gui.label_LDBP_IL_Pen_low_ival.text()))))
                gui.label_LDBP_IL_Pen_high_dev.setText(str(abs(\
                        float(gui.label_LDBP_IL_Pen_high_rval.text())-\
                        float(gui.label_LDBP_IL_Pen_high_ival.text()))))
                gui.label_LDBP_IL_LatOff_dev.setText(str(abs(\
                        float(gui.label_LDBP_IL_LatOff_rval.text())-\
                        float(gui.label_LDBP_IL_LatOff_ival.text()))))
                gui.label_LDBP_IL_MinMax_dev.setText(str(abs(\
                        float(gui.label_LDBP_IL_MinMax_rval.text())-\
                        float(gui.label_LDBP_IL_MinMax_ival.text()))))
                gui.label_LDBP_CL_FieldSize_dev.setText(str(abs(\
                        float(gui.label_LDBP_CL_FieldSize_rval.text())-\
                        float(gui.label_LDBP_CL_FieldSize_ival.text()))))
                gui.label_LDBP_CL_Pen_low_dev.setText(str(abs(\
                        float(gui.label_LDBP_CL_Pen_low_rval.text())-\
                        float(gui.label_LDBP_CL_Pen_low_ival.text()))))
                gui.label_LDBP_CL_Pen_high_dev.setText(str(abs(\
                        float(gui.label_LDBP_CL_Pen_high_rval.text())-\
                        float(gui.label_LDBP_CL_Pen_high_ival.text()))))
                gui.label_LDBP_CL_LatOff_dev.setText(str(abs(\
                        float(gui.label_LDBP_CL_LatOff_rval.text())-\
                        float(gui.label_LDBP_CL_LatOff_ival.text()))))
```

```
            gui.label_LDBP_CL_MinMax_dev.setText(str(abs(\
                    float(gui.label_LDBP_CL_MinMax_rval.text())-\
                    float(gui.label_LDBP_CL_MinMax_ival.text()))))

# LD-values compare deviation to tolerance
   #inline
            if(float(gui.label_LDBP_IL_FieldSize_dev.text())<\
                        float(gui.label_LDBP_IL_FieldSize_tol.text())):
               gui.label_LDBP_IL_FieldSize_dev.setStyleSheet(\
                        "QLabel {Background-color:lightgreen}")
            else:
               PASS=False
               gui.label_LDBP_IL_FieldSize_dev.setStyleSheet(\
                        "QLabel {Background-color:red}")
            if(float(gui.label_LDBP_IL_Pen_low_dev.text())<\
                        float(gui.label_LDBP_IL_Pen_low_tol.text())):
               gui.label_LDBP_IL_Pen_low_dev.setStyleSheet(\
                        "QLabel {Background-color:lightgreen}")
            else: gui.label_LDBP_IL_Pen_low_dev.setStyleSheet(\
                        "QLabel {Background-color:red}")
            if(float(gui.label_LDBP_IL_Pen_high_dev.text())<\
                        float(gui.label_LDBP_IL_Pen_high_tol.text())):
               gui.label_LDBP_IL_Pen_high_dev.setStyleSheet(\
                        "QLabel {Background-color:lightgreen}")
            else:
               PASS=False
               gui.label_LDBP_IL_Pen_high_dev.setStyleSheet(\
                        "QLabel {Background-color:red}")
            if(float(gui.label_LDBP_IL_LatOff_dev.text())<\
                        float(gui.label_LDBP_IL_LatOff_tol.text())):
               gui.label_LDBP_IL_LatOff_dev.setStyleSheet(\
                        "QLabel {Background-color:lightgreen}")
            else:
               PASS=False
               gui.label_LDBP_IL_LatOff_dev.setStyleSheet(\
                        "QLabel {Background-color:red}")
            if(float(gui.label_LDBP_IL_MinMax_dev.text())<\
                        float(gui.label_LDBP_IL_MinMax_tol.text())):
               gui.label_LDBP_IL_MinMax_dev.setStyleSheet(\
                        "QLabel {Background-color:lightgreen}")
            else:
               PASS=False
               gui.label_LDBP_IL_MinMax_dev.setStyleSheet(\
                        "QLabel {Background-color:red}")
   #crossline
            if(float(gui.label_LDBP_CL_FieldSize_dev.text())<\
                        float(gui.label_LDBP_CL_FieldSize_tol.text())):
               gui.label_LDBP_CL_FieldSize_dev.setStyleSheet(\
                        "QLabel {Background-color:lightgreen}")
            else:
               PASS=False
               gui.label_LDBP_CL_FieldSize_dev.setStyleSheet(\
                        "QLabel {Background-color:red}")
            if(float(gui.label_LDBP_CL_Pen_low_dev.text())<\
                        float(gui.label_LDBP_CL_Pen_low_tol.text())):
```

```
            gui.label_LDBP_CL_Pen_low_dev.setStyleSheet(\
                    "QLabel {Background-color:lightgreen}")
        else:
            PASS=False
            gui.label_LDBP_CL_Pen_low_dev.setStyleSheet(\
                    "QLabel {Background-color:red}")
        if(float(gui.label_LDBP_CL_Pen_high_dev.text())<\
                    float(gui.label_LDBP_CL_Pen_high_tol.text())):
            gui.label_LDBP_CL_Pen_high_dev.setStyleSheet(\
                    "QLabel {Background-color:lightgreen}")
        else:
            PASS=False
            gui.label_LDBP_CL_Pen_high_dev.setStyleSheet(\
                    "QLabel {Background-color:red}")
        if(float(gui.label_LDBP_CL_LatOff_dev.text())<\
                    float(gui.label_LDBP_CL_LatOff_tol.text())):
            gui.label_LDBP_CL_LatOff_dev.setStyleSheet(\
                    "QLabel {Background-color:lightgreen}")
        else:
            PASS=False
            gui.label_LDBP_CL_LatOff_dev.setStyleSheet(\
                    "QLabel {Background-color:red}")
        if(float(gui.label_LDBP_CL_MinMax_dev.text())<\
                    float(gui.label_LDBP_CL_MinMax_tol.text())):
            gui.label_LDBP_CL_MinMax_dev.setStyleSheet(\
                    "QLabel {Background-color:lightgreen}")
        else:
            PASS=False
            gui.label_LDBP_CL_MinMax_dev.setStyleSheet(\
                    "QLabel {Background-color:red}")

        if PASS:
            gui.label_master_fail_pass.setText("PASS")
            gui.label_master_fail_pass.setStyleSheet(\
                        "QLabel {Background-color:green}")
        else:
            gui.label_master_fail_pass.setText("CAUTION!")
            gui.label_master_fail_pass.setStyleSheet(\
                        "QLabel {Background-color:orange}")

    if val:
        gui.label_LDBP_RangeModulation.setText(option+" n.a.")
        gui.label_LDBP_IL_FieldSize_ival.setText("n.a.")
        gui.label_LDBP_IL_FieldSize_tol.setText("n.a.")
        gui.label_LDBP_IL_FieldSize_dev.setText("n.a.")
        gui.label_LDBP_IL_Pen_low_ival.setText("n.a.")
        gui.label_LDBP_IL_Pen_low_tol.setText("n.a.")
        gui.label_LDBP_IL_Pen_low_dev.setText("n.a.")
        gui.label_LDBP_IL_Pen_high_ival.setText("n.a.")
        gui.label_LDBP_IL_Pen_high_tol.setText("n.a.")
        gui.label_LDBP_IL_Pen_high_dev.setText("n.a.")
        gui.label_LDBP_IL_LatOff_ival.setText("n.a.")
        gui.label_LDBP_IL_LatOff_tol.setText("n.a.")
        gui.label_LDBP_IL_LatOff_dev.setText("n.a.")
        gui.label_LDBP_IL_MinMax_ival.setText("n.a.")
```

```
            gui.label_LDBP_IL_MinMax_tol.setText("n.a.")
            gui.label_LDBP_IL_MinMax_dev.setText("n.a.")
            gui.label_LDBP_CL_FieldSize_ival.setText("n.a.")
            gui.label_LDBP_CL_FieldSize_tol.setText("n.a.")
            gui.label_LDBP_CL_FieldSize_dev.setText("n.a.")
            gui.label_LDBP_CL_Pen_low_ival.setText("n.a.")
            gui.label_LDBP_CL_Pen_low_tol.setText("n.a.")
            gui.label_LDBP_CL_Pen_low_dev.setText("n.a.")
            gui.label_LDBP_CL_Pen_high_ival.setText("n.a.")
            gui.label_LDBP_CL_Pen_high_tol.setText("n.a.")
            gui.label_LDBP_CL_Pen_high_dev.setText("n.a.")
            gui.label_LDBP_CL_LatOff_ival.setText("n.a.")
            gui.label_LDBP_CL_LatOff_tol.setText("n.a.")
            gui.label_LDBP_CL_LatOff_dev.setText("n.a.")
            gui.label_LDBP_CL_MinMax_ival.setText("n.a.")
            gui.label_LDBP_CL_MinMax_tol.setText("n.a.")
            gui.label_LDBP_CL_MinMax_dev.setText("n.a.")

            gui.label_LDBP_IL_FieldSize_dev.setStyleSheet(\
                              "QLabel {Background-color:none}")
            gui.label_LDBP_IL_Pen_low_dev.setStyleSheet(\
                              "QLabel {Background-color:none}")
            gui.label_LDBP_IL_Pen_high_dev.setStyleSheet(\
                              "QLabel {Background-color:none}")
            gui.label_LDBP_IL_LatOff_dev.setStyleSheet(\
                              "QLabel {Background-color:none}")
            gui.label_LDBP_IL_MinMax_dev.setStyleSheet(\
                              "QLabel {Background-color:none}")
            gui.label_LDBP_CL_FieldSize_dev.setStyleSheet(\
                              "QLabel {Background-color:none}")
            gui.label_LDBP_CL_Pen_low_dev.setStyleSheet(\
                              "QLabel {Background-color:none}")
            gui.label_LDBP_CL_Pen_high_dev.setStyleSheet(\
                              "QLabel {Background-color:none}")
            gui.label_LDBP_CL_LatOff_dev.setStyleSheet(\
                              "QLabel {Background-color:none}")
            gui.label_LDBP_CL_MinMax_dev.setStyleSheet(\
                              "QLabel {Background-color:none}")

#-------------------------------------------------------------------------------
    def LoadDDBPIV(self,option, refFile):                              # ready
#("Load Depth Dose Blue Phantom 'Ideal' Values")
#       o option: String of the option, which has to be loaded
#
#•      load text of file "Depth Dose - Water Phantom.txt" and
#                  split it by tabs
#•      show reference values, tolerance values und deviation between
#                  measured values and reference values on the GUI
#•      compare deviation values to tolerance values,
#                  color them red or lightgreen
#•      if option not loadable, all values(except the measured values)
#                  get the string "n.a."

        gui.textEdit_MessageField.setText("")
        gui.lineEdit_DDBP_option.setText("")
```

```
        print"LoadDDWPIV"
        liste =loadtxt(refFile, delimiter = '\t', skiprows=2, dtype = str)
        val=True
        for i in range(len(liste)):
            if option==liste[i][0]:
                gui.label_master_fail_pass.setHidden(False)
                gui.lineEdit_DDBP_option.setText("")
                val=False
                PASS=True
                gui.label_DDBP_RangeModulation.setText(option)
                gui.label_DDBP_pR90_ival.setText(liste[i][1])
                gui.label_DDBP_pR90_tol.setText(liste[i][2])
                gui.label_DDBP_pR95_ival.setText(liste[i][3])
                gui.label_DDBP_pR95_tol.setText(liste[i][4])
                gui.label_DDBP_pR98_ival.setText(liste[i][5])
                gui.label_DDBP_pR98_tol.setText(liste[i][6])
                gui.label_DDBP_dR10_ival.setText(liste[i][7])
                gui.label_DDBP_dR10_tol.setText(liste[i][8])
                gui.label_DDBP_dR20_ival.setText(liste[i][9])
                gui.label_DDBP_dR20_tol.setText(liste[i][10])
                gui.label_DDBP_dR50_ival.setText(liste[i][11])
                gui.label_DDBP_dR50_tol.setText(liste[i][12])
                gui.label_DDBP_dR80_ival.setText(liste[i][13])
                gui.label_DDBP_dR80_tol.setText(liste[i][14])
                gui.label_DDBP_dR90_ival.setText(liste[i][15])
                gui.label_DDBP_dR90_tol.setText(liste[i][16])
                gui.label_DDBP_dR95_ival.setText(liste[i][17])
                gui.label_DDBP_dR95_tol.setText(liste[i][18])
                gui.label_DDBP_SOBP_9090_ival.setText(liste[i][19])
                gui.label_DDBP_SOBP_9090_tol.setText(liste[i][20])
                gui.label_DDBP_SOBP_9890_ival.setText(liste[i][21])
                gui.label_DDBP_SOBP_9890_tol.setText(liste[i][22])
                gui.label_DDBP_DFO_ival.setText(liste[i][23])
                gui.label_DDBP_DFO_tol.setText(liste[i][24])
                gui.label_DDBP_MinMax_ival.setText(liste[i][25])
                gui.label_DDBP_MinMax_tol.setText(liste[i][26])

# DD_BP-values calculate deviation
                gui.label_DDBP_pR90_dev.setText(str(abs(\
                        float(gui.label_DDBP_pR90_rval.text()) -\
                        float(gui.label_DDBP_pR90_ival.text()))))
                gui.label_DDBP_pR95_dev.setText(str(abs(\
                        float(gui.label_DDBP_pR95_rval.text()) - \
                        float(gui.label_DDBP_pR95_ival.text()))))
                gui.label_DDBP_pR98_dev.setText(str(abs(\
                        float(gui.label_DDBP_pR98_rval.text()) - \
                        float(gui.label_DDBP_pR98_ival.text()))))
                gui.label_DDBP_dR10_dev.setText(str(abs(\
                        float(gui.label_DDBP_dR10_rval.text()) - \
                        float(gui.label_DDBP_dR10_ival.text()))))
                gui.label_DDBP_dR20_dev.setText(str(abs(\
                        float(gui.label_DDBP_dR20_rval.text()) - \
                        float(gui.label_DDBP_dR20_ival.text()))))
                gui.label_DDBP_dR50_dev.setText(str(abs(\
                        float(gui.label_DDBP_dR50_rval.text()) - \
```

```
                    float(gui.label_DDBP_dR50_ival.text())))
            gui.label_DDBP_dR80_dev.setText(str(abs(\
                    float(gui.label_DDBP_dR80_rval.text()) - \
                    float(gui.label_DDBP_dR80_ival.text()))))
            gui.label_DDBP_dR90_dev.setText(str(abs(\
                    float(gui.label_DDBP_dR90_rval.text()) - \
                    float(gui.label_DDBP_dR90_ival.text()))))
            gui.label_DDBP_dR95_dev.setText(str(abs(\
                    float(gui.label_DDBP_dR95_rval.text()) - \
                    float(gui.label_DDBP_dR95_ival.text()))))
            gui.label_DDBP_SOBP_9090_dev.setText(str(abs(\
                    float(gui.label_DDBP_SOBP_9090_rval.text()) - \
                    float(gui.label_DDBP_SOBP_9090_ival.text()))))
            gui.label_DDBP_SOBP_9890_dev.setText(str(abs(\
                    float(gui.label_DDBP_SOBP_9890_rval.text()) - \
                    float(gui.label_DDBP_SOBP_9890_ival.text())) )
            gui.label_DDBP_DFO_dev.setText(str(abs(\
                    float(gui.label_DDBP_DFO_rval.text()) - \
                    float(gui.label_DDBP_DFO_ival.text()))))
            gui.label_DDBP_MinMax_dev.setText(str(abs(\
                    float(gui.label_DDBP_MinMax_rval.text()) - \
                    float(gui.label_DDBP_MinMax_ival.text()))))

# DD_BP-values compare deviation to tolerance
            if(float(gui.label_DDBP_pR90_dev.text())<\
                    float(gui.label_DDBP_pR90_tol.text())):
               gui.label_DDBP_pR90_dev.setStyleSheet(\
                    "QLabel {Background-color:lightgreen}")
            else:
               PASS=False
               gui.label_DDBP_pR90_dev.setStyleSheet(\
                    "QLabel {Background-color:red}")
            if(float(gui.label_DDBP_pR95_dev.text())<\
                    float(gui.label_DDBP_pR95_tol.text())):
               gui.label_DDBP_pR95_dev.setStyleSheet(\
                    "QLabel {Background-color:lightgreen}")
            else:
               PASS=False
               gui.label_DDBP_pR95_dev.setStyleSheet(\
                    "QLabel {Background-color:red}")
            if(float(gui.label_DDBP_pR98_dev.text())<\
                    float(gui.label_DDBP_pR98_tol.text())):
               gui.label_DDBP_pR98_dev.setStyleSheet(\
                    "QLabel {Background-color:lightgreen}")
            else:
               PASS=False
               gui.label_DDBP_pR98_dev.setStyleSheet(\
                    "QLabel {Background-color:red}")
            if(float(gui.label_DDBP_dR10_dev.text())<\
                    float(gui.label_DDBP_dR10_tol.text())):
               gui.label_DDBP_dR10_dev.setStyleSheet(\
                    "QLabel {Background-color:lightgreen}")
            else:
               PASS=False
               gui.label_DDBP_dR10_dev.setStyleSheet(\
```

```
                "QLabel {Background-color:red}")
if(float(gui.label_DDBP_dR20_dev.text())<\
                float(gui.label_DDBP_dR20_tol.text())):
   gui.label_DDBP_dR20_dev.setStyleSheet(\
                "QLabel {Background-color:lightgreen}")
else:
   PASS=False
   gui.label_DDBP_dR20_dev.setStyleSheet(\
                "QLabel {Background-color:red}")
if(float(gui.label_DDBP_dR50_dev.text())<\
                float(gui.label_DDBP_dR50_tol.text())):
   gui.label_DDBP_dR50_dev.setStyleSheet(\
                "QLabel {Background-color:lightgreen}")
else:
   PASS=False
   gui.label_DDBP_dR50_dev.setStyleSheet(\
                "QLabel {Background-color:red}")
if(float(gui.label_DDBP_dR80_dev.text())<\
                float(gui.label_DDBP_dR80_tol.text())):
   gui.label_DDBP_dR80_dev.setStyleSheet(\
                "QLabel {Background-color:lightgreen}")
else:
   PASS=False
   gui.label_DDBP_dR80_dev.setStyleSheet(\
                "QLabel {Background-color:red}")
if(float(gui.label_DDBP_dR90_dev.text())<\
                float(gui.label_DDBP_dR90_tol.text())):
   gui.label_DDBP_dR90_dev.setStyleSheet(\
                "QLabel {Background-color:lightgreen}")
else:
   PASS=False
   gui.label_DDBP_dR90_dev.setStyleSheet(\
                "QLabel {Background-color:red}")
if(float(gui.label_DDBP_dR95_dev.text())<\
                float(gui.label_DDBP_dR95_tol.text())):
   gui.label_DDBP_dR95_dev.setStyleSheet(\
                "QLabel {Background-color:lightgreen}")
else:
   PASS=False
   gui.label_DDBP_dR95_dev.setStyleSheet(\
                "QLabel {Background-color:red}")
if(float(gui.label_DDBP_SOBP_9090_dev.text())<\
                float(gui.label_DDBP_SOBP_9090_tol.text())):
   gui.label_DDBP_SOBP_9090_dev.setStyleSheet(\
                "QLabel {Background-color:lightgreen}")
else:
   PASS=False
   gui.label_DDBP_SOBP_9090_dev.setStyleSheet(\
                "QLabel {Background-color:red}")
if(float(gui.label_DDBP_SOBP_9890_dev.text())<\
                float(gui.label_DDBP_SOBP_9890_tol.text())):
   gui.label_DDBP_SOBP_9890_dev.setStyleSheet(\
                "QLabel {Background-color:lightgreen}")
else:
   PASS=False
```

```
            gui.label_DDBP_SOBP_9890_dev.setStyleSheet(\
                    "QLabel {Background-color:red}")
        if(float(gui.label_DDBP_DFO_dev.text())<\
                    float(gui.label_DDBP_DFO_tol.text())):
            gui.label_DDBP_DFO_dev.setStyleSheet(\
                    "QLabel {Background-color:lightgreen}")
        else:
            PASS=False
            gui.label_DDBP_DFO_dev.setStyleSheet(\
                    "QLabel {Background-color:red}")
        if(float(gui.label_DDBP_MinMax_dev.text())<\
                    float(gui.label_DDBP_MinMax_tol.text())):
            gui.label_DDBP_MinMax_dev.setStyleSheet(\
                    "QLabel {Background-color:lightgreen}")
        else:
            PASS=False
            gui.label_DDBP_MinMax_dev.setStyleSheet(\
                    "QLabel {Background-color:red}")

        if PASS:
            gui.label_master_fail_pass.setText("PASS")
            gui.label_master_fail_pass.setStyleSheet(\
                        "QLabel {Background-color:green}")
        else:
            gui.label_master_fail_pass.setText("CAUTION!")
            gui.label_master_fail_pass.setStyleSheet(\
                        "QLabel {Background-color:orange}")

    if val:
        gui.label_DDBP_RangeModulation.setText(option+" n.a.")
        gui.label_DDBP_pR90_ival.setText("n.a.")
        gui.label_DDBP_pR90_tol.setText("n.a.")
        gui.label_DDBP_pR90_dev.setText("n.a.")
        gui.label_DDBP_pR95_ival.setText("n.a.")
        gui.label_DDBP_pR95_tol.setText("n.a.")
        gui.label_DDBP_pR95_dev.setText("n.a.")
        gui.label_DDBP_pR98_ival.setText("n.a.")
        gui.label_DDBP_pR98_tol.setText("n.a.")
        gui.label_DDBP_pR98_dev.setText("n.a.")
        gui.label_DDBP_dR10_ival.setText("n.a.")
        gui.label_DDBP_dR10_tol.setText("n.a.")
        gui.label_DDBP_dR10_dev.setText("n.a.")
        gui.label_DDBP_dR20_ival.setText("n.a.")
        gui.label_DDBP_dR20_tol.setText("n.a.")
        gui.label_DDBP_dR20_dev.setText("n.a.")
        gui.label_DDBP_dR50_ival.setText("n.a.")
        gui.label_DDBP_dR50_tol.setText("n.a.")
        gui.label_DDBP_dR50_dev.setText("n.a.")
        gui.label_DDBP_dR80_ival.setText("n.a.")
        gui.label_DDBP_dR80_tol.setText("n.a.")
        gui.label_DDBP_dR80_dev.setText("n.a.")
        gui.label_DDBP_dR90_ival.setText("n.a.")
        gui.label_DDBP_dR90_tol.setText("n.a.")
        gui.label_DDBP_dR90_dev.setText("n.a.")
        gui.label_DDBP_dR95_ival.setText("n.a.")
```

```
            gui.label_DDBP_dR95_tol.setText("n.a.")
            gui.label_DDBP_dR95_dev.setText("n.a.")
            gui.label_DDBP_SOBP_9090_ival.setText("n.a.")
            gui.label_DDBP_SOBP_9090_tol.setText("n.a.")
            gui.label_DDBP_SOBP_9090_dev.setText("n.a.")
            gui.label_DDBP_SOBP_9890_ival.setText("n.a.")
            gui.label_DDBP_SOBP_9890_tol.setText("n.a.")
            gui.label_DDBP_SOBP_9890_dev.setText("n.a.")
            gui.label_DDBP_DFO_ival.setText("n.a.")
            gui.label_DDBP_DFO_tol.setText("n.a.")
            gui.label_DDBP_DFO_dev.setText("n.a.")
            gui.label_DDBP_MinMax_ival.setText("n.a.")
            gui.label_DDBP_MinMax_tol.setText("n.a.")
            gui.label_DDBP_MinMax_dev.setText("n.a.")
            gui.textEdit_MessageField.setText("Requested option "
                                        +option+" is not available")

            gui.label_DDBP_pR90_dev.setStyleSheet(\
                                        "QLabel {Background-color:none}")
            gui.label_DDBP_pR95_dev.setStyleSheet(\
                                        "QLabel {Background-color:none}")
            gui.label_DDBP_pR98_dev.setStyleSheet(\
                                        "QLabel {Background-color:none}")
            gui.label_DDBP_dR10_dev.setStyleSheet(\
                                        "QLabel {Background-color:none}")
            gui.label_DDBP_dR20_dev.setStyleSheet(\
                                        "QLabel {Background-color:none}")
            gui.label_DDBP_dR50_dev.setStyleSheet(\
                                        "QLabel {Background-color:none}")
            gui.label_DDBP_dR80_dev.setStyleSheet(\
                                        "QLabel {Background-color:none}")
            gui.label_DDBP_dR90_dev.setStyleSheet(\
                                        "QLabel {Background-color:none}")
            gui.label_DDBP_dR95_dev.setStyleSheet(\
                                        "QLabel {Background-color:none}")
            gui.label_DDBP_SOBP_9090_dev.setStyleSheet(\
                                        "QLabel {Background-color:none}")
            gui.label_DDBP_SOBP_9890_dev.setStyleSheet(\
                                        "QLabel {Background-color:none}")
            gui.label_DDBP_DFO_dev.setStyleSheet(\
                                        "QLabel {Background-color:none}")
            gui.label_DDBP_MinMax_dev.setStyleSheet(\
                                        "QLabel {Background-color:none}")

#-------------------------------------------------------------------------
    def GrayScale(self,arr):                                    # ready
#       o arr: array of number values
#
#•       normalize the array to 255 (Grayscale has 256 values: from 0 to 255)
#
#       o return:
#            □       normalized array

        print"GeyScale"
        arr_greyscale=1*arr*255/arr.max()
```

```
        return arr_greyscale
#-------------------------------------------------------------------------
    def CreateImage(self,arr):                                  # in progress/ready
#       o arr: array of numbers (best: min 0, max 255)
#
#•          create an image with the shape of the array
#•          image takes the values of the array as gray-scale
#                   value for the single pixel
#•          save image as .png-file
#
#       o return:
#               string of the saved image name/path

        print "Create Image"
        image_name="image.png"
        img=Image.new("L",(arr.shape[1],arr.shape[0]))
        draw=ImageDraw.Draw(img)
        for x in range(arr.shape[1]):
            for y in range(arr.shape[0]):
                draw.point((x,y),fill=int(arr[y,x]))
        img.save(image_name)
        return image_name
#-------------------------------------------------------------------------
    def ImageShow(self,img_path):                               #       ToDo
#       o img_path: string of the image-file which has to beshown
#
#•          create a pixmap of the image
#•          scale pixmap to the size of the label which will show the image
#•          show scaled pixmap in the label

        print "ImageShow"
        gui.label_image.setHidden(False)
        image_pixmap = QtGui.QPixmap(img_path)
        image_pixmap_scaled = image_pixmap.scaled(gui.label_image.size())
        gui.label_image.setPixmap(image_pixmap_scaled)
#-------------------------------------------------------------------------
    def NormalizeMax100(self,arr):
#       o arr: array of number values
#
#•          normalize the array so the maximum equals 100
#
#       o return:
#               normalized array

        print "NormalizeMax100"
        normalizedArr=100*arr/arr.max()
        return normalizedArr
#-------------------------------------------------------------------------
    def AvCalc(self,arr,row,col):                               # ready
#("Average Calculation")
#       o arr: array(2D) of numbers
#       o row: y-position in the array
#       o col: x-position in the array
#
#•          calculate the average value around the single point with
```

```
#                   the x- and y-coordinates
#•      the offset around this point is fixed on top of the
#                   source code (rectangular)
#
#      o return:
#           □      rounded value of the average

       print "AvCalc"
       val=0

       for x in range(col-X_DDWP_P_OFFSET/2,
                   col-X_DDWP_P_OFFSET/2+X_DDWP_P_OFFSET):
          for y in range(row-Y_DDWP_P_OFFSET/2,
                      row-Y_DDWP_P_OFFSET/2+Y_DDWP_P_OFFSET):
             val=val+arr[y,x]
       val=round(val/(Y_DDWP_P_OFFSET*X_DDWP_P_OFFSET),1)
       return val
#-------------------------------------------------------------------------
    def ExtractArray(self,arr,y0,y1,x0,x1):                    # ready
#      o arr:     2D array
#      o y0:      first row which has to be extracted
#      o y1: next row which has NOT to be extracted
#      o x0: first column which has NOT to be extracted
#      o x1: next column which has NOT to be extracted
#
#•      extract the sub-array of the given array from row y0 to y1 and
#                   the columns from x0 to x1
#•      if x- or y-coordinatas ar the same,
#                   the hole column or row will be returned
#                   the other coordinates will be ignored in that case
#
#      o return:
#           □      extracted array

       print "AxtractArray"
       if x0==x1:
          exArr=arr[:,x0]
       elif y0==y1:
          exArr=arr[y0,:]
       else:
          exArr=arr[y0:y1,x0:x1]
          if abs(y0-y1)==1:
             exArr=exArr[0,:]
          elif abs(x0-x1)==1:
             exArr=exArr[:,0]
       return exArr
#-------------------------------------------------------------------------
    def CalculateMinMax(self,prox,dist,d_norm,coords):          # ready
#("Calculate Minimum and Maximum")
#      o prox: start position
#      o dist: end position
#      o d_norm: array(1D) of normalized data (e.g. dose values)
#      o coords: array of coordinates (e.g. x-coordinates)
#
#•      indentify the coordinates, which are bigger than the start position and
```

```
#                   smaller than the end position
#•      takes the according data, which belong to the coordinates
#•      find the minimum an the maximum of this particular range
#
#     o return:
#          value of the minimum the particular range
#          value of the maximum the particular range

      print "UniformityRegion"
      p=where(coords>prox)
      d=where(coords<dist)
      arr=d_norm[p[0][0]:d[0][-1]]
      minVal=min(arr)
      maxVal=max(arr)
      return minVal, maxVal
#-------------------------------------------------------------------------
   def CalculatePos(self, location, value, doses, coords):                # ready
#("Calculate Position")
#     o location:  location where the data will be examined
#          "p" – proximal
#          "d" – distal
#          "pd" – proximal-distal
#     o value: value, its exact position will be determined
#     o doses: array of the data (has to come under and over the "value")
#     o coords: array of coordinates belong to "doses"
#
#•      identify all indices of the doses, which come over the "value"
#•      interpolate between:
#          the first (proximal) pair of values, which was identified
#            (dose & according coordinate) and the pair of values
#            one index in front
#          the last (distal) pair of values, which was identified
#            (dose & according coordinate) and the pair of values
#            one index after it
#•      calculate the coordinates, where the dose would equals the "value"
#
#     o return:
#          proximal(location="p") or distal(location="d")
#            coordinate, where doses equals "value", or the distance between this
coordinates(location="pd")

      print "CalculatePos " + str(location) + " R" + str(value)
      val=where(doses>value)

      coord1=coords[val[0][0]-1]+\
          (coords[val[0][0]]-coords[val[0][0]-1])*\
          (value-doses[val[0][0]-1])/(doses[val[0][0]]-doses[val[0][0]-1])
      coord2=coords[val[0][-1]]+\
          (coords[val[0][-1]+1]-coords[val[0][-1]])*\
          (doses[val[0][-1]]-value)/(doses[val[0][-1]]-doses[val[0][0]+1])

      if location=="p":
         return round(coord1,1)
      elif location=="d":
         return round(coord2,1)
```

```
        elif location == "pd":
            return abs(round(coord2-coord1,1))
#-------------------------------------------------------------------------
    def CalculatePenumbra(self,location,doses,coords):                  # ready
#       o location: location where the data will be examined
#           □       “p” – proximal
#           □       “d” – distal
#       o d_norm: array(1D) dose values
#       o coords: array of coordinates according to the doses
#
#•      locate the coordinate at the given location, where doses equals 80
#•      locate the coordinate at the given location, where doses equals 20
#
#       o return:
#           □       distance between the coordinates

        print "CalculatePenumbra "+ location

        coord_80=gui.CalculatePos(location,80,doses,coords)
        coord_20=gui.CalculatePos(location,20,doses,coords)

        result=abs(round(coord_80-coord_20,1))
        return result
#-------------------------------------------------------------------------
    def CalculateLateralOffset(self,doses,coords):                      # ready
#       o d_norm: array(1D) of lateral dose values
#       o coords: array of coordinates according to the doses
#
#•      locate the coordinate, where doses equals 50 first time (negative value)
#•      locate the coordinate, where doses equals 50 last time (positive value)
#•      summarize both coordinates and divide the result by 2
#
#       o return:
#           □       rounded result of the shift from the null-location

        print "CalculateLateralOffset"

        coord_p=gui.CalculatePos("p",50,doses,coords)
        coord_d=gui.CalculatePos("d",50,doses,coords)

        result= round((coord_p+coord_d)/2.0,1)

        return result
#-------------------------------------------------------------------------
    def CalculateDDBPVal(self, d_norm, coords):                         # ready
#(“Calculate Depth Dose Blue Phantom Values”)
#       o d_norm: normalized array(1D) of depth dose values
#       o coords: array of coordinates according to the doses
#
#•      calculate all values belonging to the depth dose analysis
#               with the water phantom
#•      show the measured values on the GUI
#           □       prox R90
#           □       prox R95
#           □       prox R98
```

```
#           dist R10
#           dist R20
#           dist R50
#           dist R80
#           dist R90
#           dist R95
#           SOBP: prox R90 – dist R90
#           SOBP: prox R98 – dist R90
#           DFO
#           Uniforminty: (Max-Min)/Max

        print "CalculateDDBPVal"
        pR90=gui.CalculatePos("p",90,d_norm,coords)
        gui.label_DDBP_pR90_rval.setText(str(pR90))
        pR95=gui.CalculatePos("p",95,d_norm,coords)
        gui.label_DDBP_pR95_rval.setText(str(pR95))
        pR98=gui.CalculatePos("p",98,d_norm,coords)
        gui.label_DDBP_pR98_rval.setText(str(pR98))

        dR10=gui.CalculatePos("d",10,d_norm,coords)
        gui.label_DDBP_dR10_rval.setText(str(dR10))
        dR20=gui.CalculatePos("d",20,d_norm,coords)
        gui.label_DDBP_dR20_rval.setText(str(dR20))
        dR50=gui.CalculatePos("d",50,d_norm,coords)
        gui.label_DDBP_dR50_rval.setText(str(dR50))
        dR80=gui.CalculatePos("d",80,d_norm,coords)
        gui.label_DDBP_dR80_rval.setText(str(dR80))
        dR90=gui.CalculatePos("d",90,d_norm,coords)
        gui.label_DDBP_dR90_rval.setText(str(dR90))
        dR95=gui.CalculatePos("d",95,d_norm,coords)
        gui.label_DDBP_dR95_rval.setText(str(dR95))

        sobp9090=gui.CalculatePos("pd",90,d_norm,coords)
        gui.label_DDBP_SOBP_9090_rval.setText(str(sobp9090))
        sobp9890=dR90-pR98
        gui.label_DDBP_SOBP_9890_rval.setText(str(sobp9890))

        dfo=dR20-dR80
        gui.label_DDBP_DFO_rval.setText(str(dfo))

        minVal, maxVal = gui.CalculateMinMax(pR98+dfo,dR95-dfo,d_norm, coords)
        minMax=(amax(d_norm)-minVal)/amax(d_norm)
        gui.label_DDBP_MinMax_rval.setText(str(round(minMax,4)))
#-------------------------------------------------------------------------------
    def Inline(self,x_arr,y_arr):                                   # ready
#       o x_arr: array of x-coordinates
#       o y_arr: array of y-coordinates
#
#•       compare maximum and minimum of x_arr and y_arr
#•       decide if coordinates strikingly differ and so if it is a inline
#                   (y is constant) scan or crossline scan(x is constant)
#
#       o return:
#           Decision:
#             • 0 – neither nor
```

```
#           •  1 – inline
#           •  2 – crossline

        print "Inline?"
        k=3
        xmax=max(x_arr)
        xmin=min(x_arr)
        ymax=max(y_arr)
        ymin=min(y_arr)
        if xmax-xmin>COORD_DEVIATION: k=k-2
        if ymax-ymin>COORD_DEVIATION: k=k-1
        return k
#---------------------------------------------------------------------------
    def CalculateLDBPVal(self, x_il, d_il, y_cl, d_cl):                 #   in progress
#("Calculate Lateral Dose Blue Phantom Values")
#       o x_il: array of coordinates for the inline calculation
#                   (moving in x direction)
#       o d_il: array of doses for the inline calculation
#                   (moving in x direction)
#       o y_cl: array of coordinates for the crossline calculation
#                   (moving in y direction)
#       o d_cl: array of doses for the crossline calculation
#                   (moving in y direction)
#
#•       calculate all values belonging to the lateral dose analysis
#                    with the water phantom
#•       show the measured values on the GUI
#           □   field size
#           □   penumbra low
#           □   penumbra high
#           □   lateral offset
#           □   uniformity region
#           □   uniformity: (Max-Min)/Max

        print "CalculateLDBPVal"
        fieldSize_il=gui.CalculatePos("pd",50,d_il,x_il)
        fieldSize_cl=gui.CalculatePos("pd",50,d_cl,y_cl)
        gui.label_LDBP_IL_FieldSize_rval.setText(str(fieldSize_il))
        gui.label_LDBP_CL_FieldSize_rval.setText(str(fieldSize_cl))

        x_low=gui.CalculatePenumbra("p",d_il,x_il)
        x_high=gui.CalculatePenumbra("d",d_il,x_il)
        gui.label_LDBP_IL_Pen_low_rval.setText(str(x_low))
        gui.label_LDBP_IL_Pen_high_rval.setText(str(x_high))

        y_low=gui.CalculatePenumbra("p",d_cl,y_cl)
        y_high=gui.CalculatePenumbra("d",d_cl,y_cl)
        gui.label_LDBP_CL_Pen_low_rval.setText(str(y_low))
        gui.label_LDBP_CL_Pen_high_rval.setText(str(y_high))

        latOffset_il=gui.CalculateLateralOffset(d_il, x_il)
        latOffset_cl=gui.CalculateLateralOffset(d_cl, y_cl)
        gui.label_LDBP_IL_LatOff_rval.setText(str(latOffset_il))
        gui.label_LDBP_CL_LatOff_rval.setText(str(latOffset_cl))
```

```
        uniformity_il_coord_p=gui.CalculatePos("p",50,d_il,x_il)+\
                                    (x_low+x_high)/2
        uniformity_il_coord_d=gui.CalculatePos("d",50,d_il,x_il)-\
                                    (x_low+x_high)/2
        minVal_il, maxVal_il=gui.CalculateMinMax(uniformity_il_coord_p,
                                     uniformity_il_coord_d,
                                     d_il,x_il)
        gui.label_LDBP_IL_UnifReg_rval.setText("["+str(uniformity_il_coord_p)+\
                                    ";"+\
                                    str(uniformity_il_coord_d)+"]")
        minMax_il=(amax(d_il)-minVal_il)/amax(d_il)
        gui.label_LDBP_IL_MinMax_rval.setText(str(round(minMax_il,4)))

        uniformity_cl_coord_p=gui.CalculatePos("p",50,d_cl,y_cl)+\
                                    (y_low+y_high)/2
        uniformity_cl_coord_d=gui.CalculatePos("d",50,d_cl,y_cl)-\
                                    (y_low+y_high)/2
        print gui.CalculatePos("p",50,d_cl,y_cl)
        print gui.CalculatePos("d",50,d_cl,y_cl)
        minVal_cl, maxVal_cl=gui.CalculateMinMax(uniformity_cl_coord_p,
                                     uniformity_cl_coord_d,
                                     d_cl,y_cl)

        gui.label_LDBP_CL_UnifReg_rval.setText("["+str(uniformity_cl_coord_p)+\
                                    ";"+\
                                    str(uniformity_cl_coord_d)+"]")
        minMax_cl=(amax(d_cl)-minVal_cl)/amax(d_cl)
        gui.label_LDBP_CL_MinMax_rval.setText(str(round(minMax_cl,4)))
#-------------------------------------------------------------------------------
    def CalculatePscans(self,var,arr):                          # ready
#       o var: type of option (lower or higher modulation)
#            "a" – P-Scans 1&3
#            "b" – P-Scans 2&4
#       o arr: array(2D) of the dose distribution
#
#•      calculate the average around fixed points
#•      show 3 representative values for each scan on the GUI

        print "CalculateP"
        if var=="a":
            p2="-"
            p4="-"
            p1=str(gui.AvCalc(arr,Y_DDWP_P1_1,X_DDWP_P1))+"/"+\
               str(gui.AvCalc(arr,Y_DDWP_P1_2,X_DDWP_P1))+"/"+\
               str(gui.AvCalc(arr,Y_DDWP_P1_3,X_DDWP_P1))
            p3=str(gui.AvCalc(arr,Y_DDWP_P3,X_DDWP_P3_1))+"/"+\
               str(gui.AvCalc(arr,Y_DDWP_P3,X_DDWP_P3_2))+"/"+\
               str(gui.AvCalc(arr,Y_DDWP_P3,X_DDWP_P3_3))
        elif var=="b":
            p1="-"
            p3="-"
            p2=str(gui.AvCalc(arr,Y_DDWP_P2,X_DDWP_P2_1))+"/"+\
               str(gui.AvCalc(arr,Y_DDWP_P2,X_DDWP_P2_2))+"/"+\
               str(gui.AvCalc(arr,Y_DDWP_P2,X_DDWP_P2_3))
            p4=str(gui.AvCalc(arr,Y_DDWP_P4_1,X_DDWP_P4))+"/"+\
```

```
            str(gui.AvCalc(arr,Y_DDWP_P4_2,X_DDWP_P4))+"/"+\
            str(gui.AvCalc(arr,Y_DDWP_P4_3,X_DDWP_P4))
        else:
            p1=str(gui.AvCalc(arr,Y_DDWP_P1_1,X_DDWP_P1))+"/"+\
              str(gui.AvCalc(arr,Y_DDWP_P1_2,X_DDWP_P1))+"/"+\
              str(gui.AvCalc(arr,Y_DDWP_P1_3,X_DDWP_P1))
            p2=str(gui.AvCalc(arr,Y_DDWP_P2,X_DDWP_P2_1))+"/"+\
              str(gui.AvCalc(arr,Y_DDWP_P2,X_DDWP_P2_2))+"/"+\
              str(gui.AvCalc(arr,Y_DDWP_P2,X_DDWP_P2_3))
            p3=str(gui.AvCalc(arr,Y_DDWP_P3,X_DDWP_P3_1))+"/"+\
              str(gui.AvCalc(arr,Y_DDWP_P3,X_DDWP_P3_2))+"/"+\
              str(gui.AvCalc(arr,Y_DDWP_P3,X_DDWP_P3_3))
            p4=str(gui.AvCalc(arr,Y_DDWP_P4_1,X_DDWP_P4))+"/"+\
              str(gui.AvCalc(arr,Y_DDWP_P4_2,X_DDWP_P4))+"/"+\
              str(gui.AvCalc(arr,Y_DDWP_P4_3,X_DDWP_P4))
            gui.textEdit_MessageField.setText("Option is unclear."+\
                                    " All P-Scans are calculated.")

        gui.label_DDWP_P1_rval.setText(str(p1))
        gui.label_DDWP_P2_rval.setText(str(p2))
        gui.label_DDWP_P3_rval.setText(str(p3))
        gui.label_DDWP_P4_rval.setText(str(p4))

#-------------------------------------------------------------------------
    def ShowRScans(self,array_data, array_x, array_y):                  # ready
#       o array_data: array of size 600x600
#       o array_x: array of x-coordinates (600)
#       o array_y: array of y-coordinates (600)
#
#•       takes the data of the middle horizontal and
#                   the middle vertical line of the data-array and
#                   plot them to the coordinates
#•       just indexes from 100 to 500 are used – range of the wedged phantom
#•       plot them in widged_2 (small, top)

        print "Show R-Scans"
        gui.widget_1.setHidden(False)
        arr_R1=array_data[300,100:500]
        arr_R2=array_data[100:500,300]

        arr_R1_x_axis=array_y[100:500]
        arr_R2_x_axis=array_x[100:500]

        gui.ShowProfil(1, arr_R1, 0, 105, "rel. Dose [%]",
                    arr_R1_x_axis, min(arr_R1_x_axis), max(arr_R1_x_axis),
                    "[mm]", "Scan R1")
        gui.ShowProfil(1, arr_R2, 0, 105, "rel. Dose [%]",
                    arr_R2_x_axis, min(arr_R2_x_axis), max(arr_R2_x_axis),
                    "[mm]", "Scan R2")
#-------------------------------------------------------------------------
    def ShowPScans(self,array_data, array_x, array_y, option):          # ready
#       o array_data: array of size 600x600
#       o array_x: array of x-coordinates (600)
#       o array_y: array of y-coordinates (600)
#       o option: type of option (lower or higher modulation)
```

```
#          □      “a” – P-Scans 1&3
#          □      “b” – P-Scans 2&4
#
#•      takes the data of given scans in the quadrants of the
#              wedged phantom and plot them to the
#              according coordinates
#•      plot them in widged_1 (small, bottom)

    print "Show P-Scans"

    gui.widget_2.canvas.axes.plot([40,40],[-10,120])
    gui.widget_2.canvas.axes.plot([80,80],[-10,120])
    gui.widget_2.canvas.axes.plot([120,120],[-10,120])
    gui.widget_2.canvas.axes.set_ylim([0,1])

    if option[2]=="a":
       arr_P1_data=array_data[100:260,X_DDWP_P1]
       arr_P1_data=arr_P1_data[::-1]
       arr_P3_data=array_data[Y_DDWP_P3,100:260]
       arr_P3_data=arr_P3_data[::-1]
       gui.ShowProfil(2, arr_P1_data, 0, 105, "rel. Dose [%]",
                arange(len(arr_P1_data)), 0, 160,
                "[mm]", "Scan P1")
       gui.ShowProfil(2, arr_P3_data, 0, 105, "rel. Dose [%]",
                arange(len(arr_P3_data)-1,-1,-1), 0, 160,
                "[mm]", "Scan P3")
    elif option[2]=="b":
       arr_P2_data=array_data[Y_DDWP_P2,340:500]
       arr_P4_data=array_data[100:260,X_DDWP_P4]
       gui.ShowProfil(2, arr_P2_data, 0, 105, "rel. Dose [%]",
                arange(len(arr_P2_data)), 0, 160,
                "[mm]", "Scan P2")
       gui.ShowProfil(2, arr_P4_data, 0, 105, "rel. Dose [%]",
                arange(len(arr_P4_data)), 0, 160,
                "[mm]", "Scan P4")
    else:
       arr_P1_data=array_data[100:260,X_DDWP_P1]
       arr_P1_data=arr_P1_data[::-1]
       arr_P2_data=array_data[Y_DDWP_P2,340:500]
       arr_P3_data=array_data[Y_DDWP_P3,100:260]
       arr_P3_data=arr_P3_data[::-1]
       arr_P4_data=array_data[100:260,X_DDWP_P4]

       gui.ShowProfil(2, arr_P1_data, 0, 105, "rel. Dose [%]",
                arange(len(arr_P1_data)), 0, 160,
                "[mm]", "Scan P1")
       gui.ShowProfil(2, arr_P2_data, 0, 105, "rel. Dose [%]",
                arange(len(arr_P2_data)), 0, 160,
                "[mm]", "Scan P2")
       gui.ShowProfil(2, arr_P3_data, 0, 105, "rel. Dose [%]",
                arange(len(arr_P3_data)-1,-1,-1), 0, 160,
                "[mm]", "Scan P3")
       gui.ShowProfil(2, arr_P4_data, 0, 105, "rel. Dose [%]",
                arange(len(arr_P4_data)), 0, 160,
                "[mm]", "Scan P4")
```

```
#-------------------------------------------------------------------------
    def CalculateDDWPVal(self,arr_norm,arr_x,arr_y,option):                 # ready
#       o array_data: array of size 600x600
#       o array_x: array of x-coordinates (600)
#       o array_y: array of y-coordinates (600)
#       o option: type of option (lower or higher modulation)
#             “a” – P-Scans 1&3
#             “b” – P-Scans 2&4
#
#•      Calculate and Show all Values belonging to the Depth Dose Analysis of the
Wedged Phantom
#       o R-Scan: Distance between 50% values
#       o P-Scan: Values of 3 representative points for each P-Scan

        print "CalculateDDWPVal"
        # R1 & R2
        arrR1=gui.ExtractArray(arr_norm,100,500,300,301)
        R1=gui.CalculatePos("pd",50,arrR1,arr_y)
        gui.label_DDWP_R1_rval.setText(str(R1))
        arrR2=gui.ExtractArray(arr_norm,300,301,100,500)
        R2=gui.CalculatePos("pd",50,arrR2,arr_x)
        gui.label_DDWP_R2_rval.setText(str(R2))
        # P1 to P4
        gui.CalculatePscans(option[2],arr_norm)

#-------------------------------------------------------------------------
    def ShowProfil(self, val , y_axis, y_min, y_max, y_text, x_axis, x_min, x_max, x_text,
text):                          # ready
#       o val: number in which widget the date will be plotted (0,1,2)
#       o y_axis: values of the y-axis
#       o y_min: minimum of the y-scale
#       o y_max: maximum of the y-scale
#       o y_text: text to the y-axis
#       o x_axis: values of the x-axis
#       o x_min: minimum of the x-scale
#       o x_max: maximum of the x-scale
#       o x_text: text to the x-axis
#       o text: text of the graph
#
#•      plot the data to the given widget
#•      scales of the axes will set to the given minimum and maximum if they are not
already in the current range

        print "ShowProfil" + str(val) +text

        if val==0:
            if gui.widget_0.canvas.axes.lines==[]:
                gui.widget_0.canvas.axes.set_xlim([x_min, x_max])
                gui.widget_0.canvas.axes.set_ylim([y_min, y_max])
            else:
                x_min_act=gui.widget_0.canvas.axes.get_xlim()[0]
                x_max_act=gui.widget_0.canvas.axes.get_xlim()[1]
                y_min_act=gui.widget_0.canvas.axes.get_ylim()[0]
                y_max_act=gui.widget_0.canvas.axes.get_ylim()[1]
```

```
            if x_min_act>x_min:
                x_min_act=x_min
                gui.widget_0.canvas.axes.set_xlim([x_min_act,x_max_act])
            if x_max_act<x_max:
                x_max_act=x_max
                gui.widget_0.canvas.axes.set_xlim([x_min_act,x_max_act])
            if y_min_act>y_min:
                y_min_act=y_min
                gui.widget_0.canvas.axes.set_ylim([y_min_act,y_max_act])
            if y_max_act<y_max:
                y_max_act=y_max
                gui.widget_0.canvas.axes.set_ylim([y_min_act,y_max_act])


        gui.widget_0.setHidden(False)
        gui.widget_0.canvas.axes.set_xlabel(x_text)
        gui.widget_0.canvas.axes.set_ylabel(y_text)
        gui.widget_0.canvas.axes.plot(x_axis,y_axis, label=text)
        gui.widget_0.canvas.axes.legend(bbox_to_anchor=
                                (0., 1.02, 1., .102), loc=3,
                                ncol=2, mode="expand",
                                borderaxespad=0.)
        gui.widget_0.canvas.draw()

    if val==1:
        if gui.widget_1.canvas.axes.lines==[]:
            gui.widget_1.canvas.axes.set_xlim([min(x_axis),max(x_axis)])
            gui.widget_1.canvas.axes.set_ylim(\
                        [min(y_axis)-0.05*abs(min(y_axis)),
                         max(y_axis)+0.05*abs(max(y_axis))])
        else:
            x_min=gui.widget_1.canvas.axes.get_xlim()[0]
            x_max=gui.widget_1.canvas.axes.get_xlim()[1]
            y_min=gui.widget_1.canvas.axes.get_ylim()[0]
            y_max=gui.widget_1.canvas.axes.get_ylim()[1]
            if x_min>min(x_axis):
                x_min=min(x_axis)
                gui.widget_1.canvas.axes.set_xlim([x_min,x_max])
            if x_max<max(x_axis):
                x_max=max(x_axis)
                gui.widget_1.canvas.axes.set_xlim([x_min,x_max])
            if y_min>min(y_axis)-0.05*abs(min(y_axis)):
                y_min=min(y_axis)-0.05*abs(min(y_axis))
                gui.widget_1.canvas.axes.set_ylim([y_min,y_max])
            if y_max<max(y_axis)+0.05*abs(max(y_axis)):
                y_max=max(y_axis)+0.05*abs(max(y_axis))
                gui.widget_1.canvas.axes.set_ylim([y_min,y_max])

        gui.widget_1.setHidden(False)
        gui.widget_1.canvas.axes.set_xlabel(x_text)
        gui.widget_1.canvas.axes.set_ylabel(y_text)
        gui.widget_1.canvas.axes.plot(x_axis,y_axis, label=text)
        gui.widget_1.canvas.axes.legend(bbox_to_anchor=
                                (0., 1.02, 1., .102), loc=3,
                                ncol=2, mode="expand",
```

```
                                borderaxespad=0.)
            gui.widget_1.canvas.draw()

        elif val==2:
            if gui.widget_2.canvas.axes.lines==[]:
                gui.widget_2.canvas.axes.set_xlim([min(x_axis),max(x_axis)])
                gui.widget_2.canvas.axes.set_ylim(\
                            [min(y_axis)-0.05*abs(min(y_axis)),
                             max(y_axis)+0.05*abs(max(y_axis))])
            else:
                x_min=gui.widget_2.canvas.axes.get_xlim()[0]
                x_max=gui.widget_2.canvas.axes.get_xlim()[1]
                y_min=gui.widget_2.canvas.axes.get_ylim()[0]
                y_max=gui.widget_2.canvas.axes.get_ylim()[1]
                if x_min>min(x_axis):
                    x_min=min(x_axis)
                    gui.widget_2.canvas.axes.set_xlim([x_min,x_max])
                if x_max<max(x_axis):
                    x_max=max(x_axis)
                    gui.widget_2.canvas.axes.set_xlim([x_min,x_max])
                if y_min>min(y_axis)-0.05*abs(min(y_axis)):
                    y_min=min(y_axis)-0.05*abs(min(y_axis))
                    gui.widget_2.canvas.axes.set_ylim([y_min,y_max])
                if y_max<max(y_axis)+0.05*abs(max(y_axis)):
                    y_max=max(y_axis)+0.05*abs(max(y_axis))
                    gui.widget_2.canvas.axes.set_ylim([y_min,y_max])

            gui.widget_2.setHidden(False)
            gui.widget_2.canvas.axes.set_xlabel(x_text)
            gui.widget_2.canvas.axes.set_ylabel(y_text)
            gui.widget_2.canvas.axes.plot(x_axis,y_axis,label=text)
            gui.widget_2.canvas.axes.legend(bbox_to_anchor=
                                    (0., 1.02, 1., .102), loc=3,
                                    ncol=2, mode="expand",
                                    borderaxespad=0.)
            gui.widget_2.canvas.draw()

#---------------------------------------------------------------------------
    def LoadOption(self):
        print "LoadOption"
        gui.label_master_fail_pass.setText("")
        gui.label_master_fail_pass.setStyleSheet(\
                                "QLabel {Background-color:none}")
        if gui.labelAnalyseMenu.text()=="Daily QA - Double Scattering - "+\
                                "Depth Dose - LYNX (Wedge Phantom)":
            option=gui.lineEdit_DDWP_option.text()
            gui.LoadDQADSDDLYNXIV(option,
                            "Depth Dose - Wedge Phantom - LYNX.txt")
        elif  gui.labelAnalyseMenu.text()=="Depth Dose Analysing - "+\
                                "Water Phantom":
            option=gui.lineEdit_DDBP_option.text()
            gui.LoadDDBPIV(option,"Depth Dose - Water Phantom.txt")
        elif gui.labelAnalyseMenu.text()=="Lateral Dose Analysing - "+\
                                "Water Phantom":
            option=gui.lineEdit_LDBP_option.text()
```

```
            gui.LoadLDBPIV(option,"Lateral Dose - Water Phantom.txt")
        elif gui.labelAnalyseMenu.text()=="Daily QA - Double Scattering"+\
                                " - Lateral Dose - LYNX (Flat Phantom)":
            option=gui.lineEdit_LDBP_option.text()
            gui.LoadLDBPIV(option,"Lateral Dose - Wedge Phantom - LYNX.txt")
#-------------------------------------------------------------------------------
    def ExportPdf(self):
#•      Save plots as jpg-images
#•      load header of the input file
#•      write header and text of every active field into a text file
#•      read text file line by line and add the lines to a “rhyme” of a pdf
#                 and write it on a page
#•      draw images to new page of the pdf file
#•      save pdf file

        print "Export Pdf"
        lt=time.localtime()
        year, month, day, hour, minute, second = lt[0:6]
        inputFile=gui.lineEdit_FilePath.text()

#Textfile Export LYNX Depth Dose
        if gui.labelAnalyseMenu.text()=="Daily QA - Double Scattering - "+\
                                "Depth Dose - LYNX (Wedge Phantom)":
            inp=open(inputFile,"r")
            lineNumber=0
            liste=[]
            while lineNumber <30:
                liste.append(inp.readline())
                print lineNumber, liste[lineNumber]
                lineNumber=lineNumber+1

            outpf_txt="Export_"+str(year)+"_"+str(month)+\
                    "_"+str(day)+"_QA_DS_DD.txt"
            outp=open("Export\\temp\\"+outpf_txt,"w")
            outpf_jpg1="Export_"+str(year)+"_"+str(month)+\
                    "_"+str(day)+"_QA_DS_DD_Diagramm_R_Scan.jpg"
            outpf_jpg2="Export_"+str(year)+"_"+str(month)+\
                    "_"+str(day)+"_QA_DS_DD_Diagramm_P_Scan.jpg"
            gui.widget_1.canvas.fig.savefig("Export\\temp\\"+outpf_jpg1)
            gui.widget_2.canvas.fig.savefig("Export\\temp\\"+outpf_jpg2)
            outp.write("Quality Assurance" +
                       "\n\nDouble Scattering" +
                       "\nDepth Dose" +
                       "\n\nLYNX 2D" +
                       "\nWedged Phantom")
            outp.write("\n\n"+liste[7]+liste[8]+liste[9])

            outp.write("Date of interpretation : "+
                    str(day)+"/"+str(month)+"/"+str(year)+
                    "\nTime of interpretation : "+
                    str(hour)+":"+str(minute)+":"+str(second)
                    )
            outp.write("\n"+liste[18])
            if gui.label_DDWP_RangeModulation.text()[-4:]!="n.a.":
                option=gui.label_DDWP_RangeModulation.text()[-4:-1]
```

```
    option_Range=gui.label_DDWP_RangeModulation.text()[\
          gui.label_DDWP_RangeModulation.text().indexOf("R")+2:\
          gui.label_DDWP_RangeModulation.text().indexOf("M")-1]
    option_Modulation=gui.label_DDWP_RangeModulation.text()[\
          gui.label_DDWP_RangeModulation.text().indexOf("M")+2:\
          gui.label_DDWP_RangeModulation.text().indexOf("(")-1]
else:
    option="unknown"
    option_Range="unknown"
    option_Modulation="unknown"
outp.write("\nOption : " + option +
          "\nRange : " + option_Range+ " cm"+
          "\nModulation : " + option_Modulation+ " cm")

outp.write("\n\n\tMeasured"+
          "\tReference"+
          "\tTolerance"+
          "\tDeviation")
outp.write("\n\tValue   "+
          "\tValue    "+
          "\tValue    "+
          "\tValue    ")

outp.write("\n\nR1[mm]: "+gui.label_DDWP_R1_rval.text()+
          "   \t"+gui.label_DDWP_R1_ival.text()+
          "    \t"+gui.label_DDWP_R1_tol.text()+
          "     \t"+gui.label_DDWP_R1_dev.text())
outp.write("\nR2[mm]: "+gui.label_DDWP_R2_rval.text()+
          "   \t"+gui.label_DDWP_R2_ival.text()+
          "    \t"+gui.label_DDWP_R2_tol.text()+
          "     \t"+gui.label_DDWP_R2_dev.text())

outp.write("\n\nP1:     "+gui.label_DDWP_P1_rval.text()+
          "  "+gui.label_DDWP_P1_ival.text()+
          "   "+gui.label_DDWP_P1_tol.text()+
          "     \t"+gui.label_DDWP_P1_dev.text())
outp.write("\nP2:     "+gui.label_DDWP_P2_rval.text()+
          "  "+gui.label_DDWP_P2_ival.text()+
          "   "+gui.label_DDWP_P2_tol.text()+
          "     \t"+gui.label_DDWP_P2_dev.text())
outp.write("\nP3:     "+gui.label_DDWP_P3_rval.text()+
          "  "+gui.label_DDWP_P3_ival.text()+
          "   "+gui.label_DDWP_P3_tol.text()+
          "     \t"+gui.label_DDWP_P3_dev.text())
outp.write("\nP4:     "+gui.label_DDWP_P4_rval.text()+
          "  "+gui.label_DDWP_P4_ival.text()+
          "   "+gui.label_DDWP_P4_tol.text()+
          "     \t"+gui.label_DDWP_P4_dev.text())
outp.close()

pdf=Canvas("Export\\"+outpf_txt[:-3]+"pdf")
pdf.setFont("Courier", 10)
rhyme = pdf.beginText(50, 800)

lineNumber=0
```

```
        savefile= open("Export\\temp\\"+outpf_txt,"r")
        liste =savefile.readlines()
        i=0
        print liste
        while i<len(liste):
           liste[i]=liste[i].replace("\n","")
           liste[i]=liste[i].replace("\t","        ")
           rhyme.textLine(liste[i])
           i=i+1
        pdf.drawText(rhyme)
        pdf.showPage()
        pdf.drawImage("Export\\temp\\"+outpf_jpg1,0,450)
        pdf.drawImage("Export\\temp\\"+outpf_jpg2,0,50)
        pdf.showPage()
        pdf.save()


#Textfile Export Water Phantom Depth Dose
    elif gui.labelAnalyseMenu.text()=="Depth Dose Analysing - "+\
                                "Water Phantom":
        outpf_txt="Export_"+str(year)+"_"+str(month)+\
                "_"+str(day)+"_Analyse_DD.txt"
        outp=open("Export\\temp\\"+outpf_txt,"w")
        outpf_jpg="Export_"+str(year)+"_"+str(month)+\
                "_"+str(day)+"_Analyse_DD_Diagramm.jpg"
        gui.widget_0.canvas.fig.savefig("Export\\temp\\"+outpf_jpg)
        outp.write("Analysing" +
                    "\nDouble Scattering" +
                    "\nDepth Dose" +
                    "\nOmniProAccept" +
                    "\nWater Phantom")
        outp.write("\n\nDate of interpretation : "+
                    str(day)+"/"+str(month)+"/"+str(year))
        if gui.label_DDBP_RangeModulation.text()[-4:]!="n.a.":
           option_Range=gui.label_DDBP_RangeModulation.text()[\
                  gui.label_DDBP_RangeModulation.text().indexOf("R")+1:\
                  gui.label_DDBP_RangeModulation.text().indexOf("M")]
           option_Modulation=gui.label_DDBP_RangeModulation.text()[\
                  gui.label_DDBP_RangeModulation.text().indexOf("M")+1:]
        else:
           option_Range="unknown"
           option_Modulation="unknown"
        outp.write("\nRange : "+ option_Range+ " mm" +
                  "\nModulation : "+ str(int(option_Modulation))+ " mm")

        outp.write("\n\n\t\tMeasured       "+
                  "Reference      "+
                  "Tolerance      "+
                  "Deviation      "+
                  "Fail/")
        outp.write("\n\t\tValue          "+
                  "Value          "+
                  "Value          "+
                  "Value          "+
                  "Pass")
```

```
outp.write("\n\nproximal"+
        "\nR90[mm]:"+
        "\t"+gui.label_DDBP_pR90_rval.text()+
        "\t    "+gui.label_DDBP_pR90_ival.text()+
        "\t    "+gui.label_DDBP_pR90_tol.text()+
        "\t   "+gui.label_DDBP_pR90_dev.text()+
        "\t    ")
outp.write("\nR95[mm]:"+
        "\t"+gui.label_DDBP_pR95_rval.text()+
        "\t    "+gui.label_DDBP_pR95_ival.text()+
        "\t    "+gui.label_DDBP_pR95_tol.text()+
        "\t   "+gui.label_DDBP_pR95_dev.text()+
        "\t   ")
outp.write("\nR98[mm]:"+
        "\t"+gui.label_DDBP_pR98_rval.text()+
        "\t    "+gui.label_DDBP_pR98_ival.text()+
        "\t    "+gui.label_DDBP_pR98_tol.text()+
        "\t   "+gui.label_DDBP_pR98_dev.text()+
        "\t   ")

outp.write("\n\ndistal"+
        "\nR10[mm]:"+
        "\t"+gui.label_DDBP_dR10_rval.text()+
        "\t    "+gui.label_DDBP_dR10_ival.text()+
        "\t    "+gui.label_DDBP_dR10_tol.text()+
        "\t   "+gui.label_DDBP_dR10_dev.text()+
        "\t   ")
outp.write("\nR20[mm]:"+
        "\t"+gui.label_DDBP_dR20_rval.text()+
        "\t    "+gui.label_DDBP_dR20_ival.text()+
        "\t    "+gui.label_DDBP_dR20_tol.text()+
        "\t   "+gui.label_DDBP_dR20_dev.text()+
        "\t   ")
outp.write("\nR50[mm]:"+
        "\t"+gui.label_DDBP_dR50_rval.text()+
        "\t    "+gui.label_DDBP_dR50_ival.text()+
        "\t    "+gui.label_DDBP_dR50_tol.text()+
        "\t   "+gui.label_DDBP_dR50_dev.text()+
        "\t   ")
outp.write("\nR80[mm]:"+
        "\t"+gui.label_DDBP_dR80_rval.text()+
        "\t    "+gui.label_DDBP_dR80_ival.text()+
        "\t    "+gui.label_DDBP_dR80_tol.text()+
        "\t   "+gui.label_DDBP_dR80_dev.text()+
        "\t   ")
outp.write("\nR90[mm]:"+
        "\t"+gui.label_DDBP_dR90_rval.text()+
        "\t    "+gui.label_DDBP_dR90_ival.text()+
        "\t    "+gui.label_DDBP_dR90_tol.text()+
        "\t   "+gui.label_DDBP_dR90_dev.text()+
        "\t   ")
outp.write("\nR95[mm]:"+
        "\t"+gui.label_DDBP_dR95_rval.text()+
        "\t    "+gui.label_DDBP_dR95_ival.text()+
```

```
            "\t    "+gui.label_DDBP_dR95_tol.text()+
            "\t   "+gui.label_DDBP_dR95_dev.text()+
            "\t   ")
outp.write("\n\nSOBP:"+
            "\npR90-dR90[mm]:  "+
            ""+gui.label_DDBP_SOBP_9090_rval.text()+
            "\t   "+gui.label_DDBP_SOBP_9090_ival.text()+
            "\t   "+gui.label_DDBP_SOBP_9090_tol.text()+
            "\t    "+gui.label_DDBP_SOBP_9090_dev.text()+
            "\t    ")
outp.write("\npR98-dR90[mm]:  "+
            ""+gui.label_DDBP_SOBP_9890_rval.text()+
            "\t   "+gui.label_DDBP_SOBP_9890_ival.text()+
            "\t   "+gui.label_DDBP_SOBP_9890_tol.text()+
            "\t    "+gui.label_DDBP_SOBP_9890_dev.text()+
            "\t    ")
outp.write("\n\nDFO:"+
            "\ndR80-dR20[mm]:  "
            ""+gui.label_DDBP_DFO_rval.text()+
            "\t    "+gui.label_DDBP_DFO_ival.text()+
            "\t    "+gui.label_DDBP_DFO_tol.text()+
            "\t    "+gui.label_DDBP_DFO_dev.text()+
            "\t    ")
outp.write("\n\nUniformity*:"+
            "\n(Max-Min)/Max:  "
            ""+gui.label_DDBP_MinMax_rval.text()+
            "\t  "+gui.label_DDBP_MinMax_ival.text()+
            "\t  "+gui.label_DDBP_MinMax_tol.text()+
            "\t  "+gui.label_DDBP_MinMax_dev.text()+
            "\t    ")

outp.write("\n\n\n\n\nUniformity:\nMin, Max in Uniformity Region"+
            "\nUnifomity Region: \n\t\tproximal R50 + 1 DFO"+
            "\n\t\tdistal R50 - 1 DFO")
outp.close()

pdf=Canvas("Export\\"+outpf_txt[:-3]+"pdf")
pdf.setFont("Courier", 10)
rhyme = pdf.beginText(50, 800)

lineNumber=0
savefile= open("Export\\temp\\"+outpf_txt,"r")
liste =savefile.readlines()
i=0
print liste
while i<len(liste):
   liste[i]=liste[i].replace("\n","")
   liste[i]=liste[i].replace("\t","        ")
   rhyme.textLine(liste[i])
   i=i+1
pdf.drawText(rhyme)
pdf.showPage()
pdf.drawImage("Export\\temp\\"+outpf_jpg,0,100)
pdf.showPage()
pdf.save()
```

```
#Textfile Export Water Phantom Lateral Dose
        elif gui.labelAnalyseMenu.text()=="Lateral Dose Analysing - "+\
                                    "Water Phantom":
            outpf_txt="Export_"+str(year)+"_"+str(month)+\
                    "_"+str(day)+"_Analyse_LD.txt"
            outp=open("Export\\temp\\"+outpf_txt,"w")
            outpf_jpg="Export_"+str(year)+"_"+str(month)+\
                    "_"+str(day)+"_Analyse_LD_Diagramm.jpg"
            gui.widget_0.canvas.fig.savefig("Export\\temp\\"+outpf_jpg)
            outp.write("Analysing" +
                       "\nDouble Scattering" +
                       "\nDepth Dose" +
                       "\nOmniProAccept" +
                       "\nWater Phantom")
            outp.write("\n\nDate of interpretation : "+
                       str(day)+"/"+str(month)+"/"+str(year))
            if gui.label_LDBP_RangeModulation.text()[-4:]!="n.a.":
               option_Range=gui.label_LDBP_RangeModulation.text()[\
                  gui.label_LDBP_RangeModulation.text().indexOf("R")+1:
                  gui.label_LDBP_RangeModulation.text().indexOf("M")]
               option_Modulation=str(int(\
                  gui.label_LDBP_RangeModulation.text()[\
                  gui.label_LDBP_RangeModulation.text().indexOf("M")+1:
                  gui.label_LDBP_RangeModulation.text().indexOf(")")]))
            else:
               option_Range="unknown"
               option_Modulation="unknown"
            outp.write("\nRange : " + option_Range+ " mm" +
                     "\nModulation : " + option_Modulation + " mm")

            outp.write("\n\n\nInline(Y):      "+
                     "X: "+gui.label_LDBP_IL_xCoord_val.text()+
                     "\n\t\tY: "+gui.label_LDBP_IL_yCoord_val.text()+
                     "\n\t\tZ: "+gui.label_LDBP_IL_zCoord_val.text()
                     )
            outp.write("\n\n\t\tMeasured        "+
                     "Reference       "+
                     "Tolerance       "+
                     "Deviation       "+
                     "Fail/")
            outp.write("\n\t\tValue           "+
                     "Value           "+
                     "Value           "+
                     "Value           "+
                     "Pass")
            outp.write("\n\nField Size[mm]: "
                     ""+ gui.label_LDBP_IL_FieldSize_rval.text()+
                     "\t   "+ gui.label_LDBP_IL_FieldSize_ival.text()+
                     "\t   "+ gui.label_LDBP_IL_FieldSize_tol.text()+
                     "\t    "+ gui.label_LDBP_IL_FieldSize_dev.text()+
                     "\t    "
                     )
            outp.write("\n\nPenumbra:       "
```

```
            "\nx_low[mm]:      "
            ""+ gui.label_LDBP_IL_Pen_low_rval.text()+
            "\t     "+ gui.label_LDBP_IL_Pen_low_ival.text()+
            "\t     "+ gui.label_LDBP_IL_Pen_low_tol.text()+
            "\t     "+ gui.label_LDBP_IL_Pen_low_dev.text()+
            "\t    "
            )
        outp.write("\nx_high[mm]:     "
            ""+ gui.label_LDBP_IL_Pen_high_rval.text()+
            "\t     "+ gui.label_LDBP_IL_Pen_high_ival.text()+
            "\t     "+ gui.label_LDBP_IL_Pen_high_tol.text()+
            "\t     "+ gui.label_LDBP_IL_Pen_high_dev.text()+
            "\t    "
            )
        outp.write("\n\nLateral\nOffset[mm]:     "
            ""+ gui.label_LDBP_IL_LatOff_rval.text()+
            "\t     "+ gui.label_LDBP_IL_LatOff_ival.text()+
            "\t     "+ gui.label_LDBP_IL_LatOff_tol.text()+
            "\t     "+ gui.label_LDBP_IL_LatOff_dev.text()+
            "\t    "
            )
        outp.write("\n\nUniformity\nRegion[mm]:     "
            ""+ gui.label_LDBP_IL_UnifReg_rval.text()
            )
        outp.write("\n(Max-Min)/Max:  "
            ""+ gui.label_LDBP_IL_MinMax_rval.text()+
            "\t   "+ gui.label_LDBP_IL_MinMax_ival.text()+
            "\t   "+ gui.label_LDBP_IL_MinMax_tol.text()+
            "\t    "+ gui.label_LDBP_IL_MinMax_dev.text()+
            "\t   "
            )

        outp.write("\n\n\nCrossline(X):   "+
            "X: "+gui.label_LDBP_CL_xCoord_val.text()+
            "\n\t\tY: "+gui.label_LDBP_CL_yCoord_val.text()+
            "\n\t\tZ: "+gui.label_LDBP_CL_zCoord_val.text()
            )
        outp.write("\n\n\t\tMeasured        "+
            "Reference       "+
            "Tolerance       "+
            "Deviation       "+
            "Fail/")
        outp.write("\n\t\tValue           "+
            "Value           "+
            "Value           "+
            "Value           "+
            "Pass")
        outp.write("\n\nField Size[mm]: "
            ""+ gui.label_LDBP_CL_FieldSize_rval.text()+
            "\t   "+ gui.label_LDBP_CL_FieldSize_ival.text()+
            "\t   "+ gui.label_LDBP_CL_FieldSize_tol.text()+
            "\t    "+ gui.label_LDBP_CL_FieldSize_dev.text()+
            "\t    "
            )
        outp.write("\n\nPenumbra:       "
```

```
                "\ny_low[mm]:      "
                ""+ gui.label_LDBP_CL_Pen_low_rval.text()+
                "\t     "+ gui.label_LDBP_CL_Pen_low_ival.text()+
                "\t     "+ gui.label_LDBP_CL_Pen_low_tol.text()+
                "\t     "+ gui.label_LDBP_CL_Pen_low_dev.text()+
                "\t    "
                )
        outp.write("\ny_high[mm]:     "
                ""+ gui.label_LDBP_CL_Pen_high_rval.text()+
                "\t     "+ gui.label_LDBP_CL_Pen_high_ival.text()+
                "\t     "+ gui.label_LDBP_CL_Pen_high_tol.text()+
                "\t     "+ gui.label_LDBP_CL_Pen_high_dev.text()+
                "\t    "
                )
        outp.write("\n\nLateral\nOffset[mm]:     "
                ""+ gui.label_LDBP_CL_LatOff_rval.text()+
                "\t     "+ gui.label_LDBP_CL_LatOff_ival.text()+
                "\t     "+ gui.label_LDBP_CL_LatOff_tol.text()+
                "\t     "+ gui.label_LDBP_CL_LatOff_dev.text()+
                "\t    "
                )
        outp.write("\n\nUniformity\nRegion[mm]:     "
                ""+ gui.label_LDBP_CL_UnifReg_rval.text()
                )
        outp.write("\n(Max-Min)/Max:  "
                ""+ gui.label_LDBP_CL_MinMax_rval.text()+
                "\t   "+ gui.label_LDBP_CL_MinMax_ival.text()+
                "\t   "+ gui.label_LDBP_CL_MinMax_tol.text()+
                "\t    "+ gui.label_LDBP_CL_MinMax_dev.text()+
                "\t   "
                )

        outp.close()

        pdf=Canvas("Export\\"+outpf_txt[:-3]+"pdf")
        pdf.setFont("Courier", 10)
        rhyme = pdf.beginText(50, 800)

        lineNumber=0
        savefile= open("Export\\temp\\"+outpf_txt,"r")
        liste =savefile.readlines()
        i=0
        print liste
        while i<len(liste):
           liste[i]=liste[i].replace("\n","")
           liste[i]=liste[i].replace("\t","        ")
           rhyme.textLine(liste[i])
           i=i+1
        pdf.drawText(rhyme)
        pdf.showPage()
        pdf.drawImage("Export\\temp\\"+outpf_jpg, 0, 100)
        pdf.showPage()
        pdf.save()

#Textfile Export LYNX Lateral Dose
```

```
elif gui.labelAnalyseMenu.text()=="Daily QA - Double Scattering"+\
                        " - Lateral Dose - LYNX (Flat Phantom)":
    inp=open(inputFile,"r")
    lineNumber=0
    liste=[]
    while lineNumber <30:
        liste.append(inp.readline())
        lineNumber=lineNumber+1
    outpf_txt="Export_"+str(year)+"_"+str(month)+\
            "_"+str(day)+"_QA_DS_LD.txt"
    outp=open("Export\\temp\\"+outpf_txt,"w")
    outpf_jpg="Export_"+str(year)+"_"+str(month)+\
            "_"+str(day)+"_QA_DS_LD_Diagramm.pdf"
    gui.widget_0.canvas.fig.savefig("Export\\temp\\"+outpf_jpg)

    outp.write("Quality Assurance" +
                "\n\nDouble Scattering" +
                "\nLateral Dose" +
                "\n\nLYNX 2D" +
                "\nWedged Phantom")
    outp.write("\n\n"+liste[7]+liste[8]+liste[9])

    outp.write("Date of interpretation : "+
            str(day)+"/"+str(month)+"/"+str(year)+
            "\nTime of interpretation : "+
            str(hour)+":"+str(minute)+":"+str(second)
            )
    outp.write("\n"+liste[18])
    if gui.label_LDBP_RangeModulation.text()[-4:]!="n.a.":
        option_Range=gui.label_LDBP_RangeModulation.text()[\
            gui.label_LDBP_RangeModulation.text().indexOf("R")+1:
            gui.label_LDBP_RangeModulation.text().indexOf("M")]
        option_Modulation=str(int(\
            gui.label_LDBP_RangeModulation.text()[\
            gui.label_LDBP_RangeModulation.text().indexOf("M")+1:
            gui.label_LDBP_RangeModulation.text().indexOf("(")]))
    else:
        option_Range="unknown"
        option_Modulation="unknown"
    outp.write("\nRange : " + option_Range+ " mm" +
            "\nModulation : " + option_Modulation + " mm")

    outp.write("\n\n\nInline(Y):      "+
            "X: "+gui.label_LDBP_IL_xCoord_val.text()+
            "\n\t\tY: "+gui.label_LDBP_IL_yCoord_val.text()+
            "\n\t\tZ: "+gui.label_LDBP_IL_zCoord_val.text()
            )
    outp.write("\n\n\t\tMeasured        "+
            "Reference       "+
            "Tolerance       "+
            "Deviation       "+
            "Fail/")
    outp.write("\n\t\tValue          "+
            "Value          "+
            "Value          "+
```

```
            "Value          "+
            "Pass")
        outp.write("\n\nField Size[mm]: "
            ""+ gui.label_LDBP_IL_FieldSize_rval.text()+
            "\t  "+ gui.label_LDBP_IL_FieldSize_ival.text()+
            "\t  "+ gui.label_LDBP_IL_FieldSize_tol.text()+
            "\t   "+ gui.label_LDBP_IL_FieldSize_dev.text()+
            "\t   "
            )
        outp.write("\n\nPenumbra:       "
            "\nx_low[mm]:      "
            ""+ gui.label_LDBP_IL_Pen_low_rval.text()+
            "\t    "+ gui.label_LDBP_IL_Pen_low_ival.text()+
            "\t    "+ gui.label_LDBP_IL_Pen_low_tol.text()+
            "\t    "+ gui.label_LDBP_IL_Pen_low_dev.text()+
            "\t   "
            )
        outp.write("\nx_high[mm]:     "
            ""+ gui.label_LDBP_IL_Pen_high_rval.text()+
            "\t    "+ gui.label_LDBP_IL_Pen_high_ival.text()+
            "\t    "+ gui.label_LDBP_IL_Pen_high_tol.text()+
            "\t    "+ gui.label_LDBP_IL_Pen_high_dev.text()+
            "\t   "
            )
        outp.write("\n\nLateral\nOffset[mm]:     "
            ""+ gui.label_LDBP_IL_LatOff_rval.text()+
            "\t    "+ gui.label_LDBP_IL_LatOff_ival.text()+
            "\t    "+ gui.label_LDBP_IL_LatOff_tol.text()+
            "\t    "+ gui.label_LDBP_IL_LatOff_dev.text()+
            "\t   "
            )
        outp.write("\n\nUniformity\nRegion[mm]:     "
            ""+ gui.label_LDBP_IL_UnifReg_rval.text()
            )
        outp.write("\n(Max-Min)/Max:  "
            ""+ gui.label_LDBP_IL_MinMax_rval.text()+
            "\t  "+ gui.label_LDBP_IL_MinMax_ival.text()+
            "\t  "+ gui.label_LDBP_IL_MinMax_tol.text()+
            "\t   "+ gui.label_LDBP_IL_MinMax_dev.text()+
            "\t  "
            )

        outp.write("\n\n\nCrossline(X):   "+
            "X: "+gui.label_LDBP_CL_xCoord_val.text()+
            "\n\t\tY: "+gui.label_LDBP_CL_yCoord_val.text()+
            "\n\t\tZ: "+gui.label_LDBP_CL_zCoord_val.text()
            )
        outp.write("\n\n\t\tMeasured        "+
            "Reference       "+
            "Tolerance       "+
            "Deviation       "+
            "Fail/")
        outp.write("\n\t\tValue          "+
            "Value          "+
            "Value          "+
```

```
        "Value          "+
        "Pass")
outp.write("\n\nField Size[mm]: "
        ""+ gui.label_LDBP_CL_FieldSize_rval.text()+
        "\t  "+ gui.label_LDBP_CL_FieldSize_ival.text()+
        "\t  "+ gui.label_LDBP_CL_FieldSize_tol.text()+
        "\t   "+ gui.label_LDBP_CL_FieldSize_dev.text()+
        "\t   "
        )
outp.write("\n\nPenumbra:       "
        "\ny_low[mm]:      "
        ""+ gui.label_LDBP_CL_Pen_low_rval.text()+
        "\t    "+ gui.label_LDBP_CL_Pen_low_ival.text()+
        "\t    "+ gui.label_LDBP_CL_Pen_low_tol.text()+
        "\t    "+ gui.label_LDBP_CL_Pen_low_dev.text()+
        "\t   "
        )
outp.write("\ny_high[mm]:     "
        ""+ gui.label_LDBP_CL_Pen_high_rval.text()+
        "\t    "+ gui.label_LDBP_CL_Pen_high_ival.text()+
        "\t    "+ gui.label_LDBP_CL_Pen_high_tol.text()+
        "\t    "+ gui.label_LDBP_CL_Pen_high_dev.text()+
        "\t   "
        )
outp.write("\n\nLateral\nOffset[mm]:     "
        ""+ gui.label_LDBP_CL_LatOff_rval.text()+
        "\t    "+ gui.label_LDBP_CL_LatOff_ival.text()+
        "\t    "+ gui.label_LDBP_CL_LatOff_tol.text()+
        "\t    "+ gui.label_LDBP_CL_LatOff_dev.text()+
        "\t   "
        )
outp.write("\n\nUniformity\nRegion[mm]:     "
        ""+ gui.label_LDBP_CL_UnifReg_rval.text()
        )
outp.write("\n(Max-Min)/Max:  "
        ""+ gui.label_LDBP_CL_MinMax_rval.text()+
        "\t  "+ gui.label_LDBP_CL_MinMax_ival.text()+
        "\t  "+ gui.label_LDBP_CL_MinMax_tol.text()+
        "\t   "+ gui.label_LDBP_CL_MinMax_dev.text()+
        "\t  "
        )

outp.close()

pdf=Canvas("Export\\"+outpf_txt[:-3]+"pdf")
pdf.setFont("Courier", 10)
rhyme = pdf.beginText(50, 800)

lineNumber=0
savefile= open("Export\\temp\\"+outpf_txt,"r")
liste =savefile.readlines()
i=0
print liste
while i<len(liste):
   liste[i]=liste[i].replace("\n","")
```

```
            liste[i]=liste[i].replace("\t","        ")
            rhyme.textLine(liste[i])
            i=i+1
        pdf.drawText(rhyme)
        pdf.showPage()
        pdf.drawImage("Export\\temp\\"+outpf_jpg, 0, 100)
        pdf.showPage()
        pdf.save()
#-------------------------------------------------------------------------
            ############### Diffrent Analyses ###############             #

    def WaterPhantomDepthDose(self, checked=None):                          # ready
#•      set the group box for depth dose measurements with the water phantom (blue
phantom) visible
#•      set text of heading
#•      ask for file to open
#•      extract option(range & modulation) from the input filename (first 8 letter)
#•      load coordinates and doses of the first measurement series
#•      normalize the doses to the maximum
#•      calculate all values that belong to the measurement
#       o prox R90
#       o prox R95
#       o prox R98
#       o dist R10
#       o dist R20
#       o dist R50
#       o dist R80
#       o dist R90
#       o dist R95
#       o SOBP: prox R90 – dist R90
#       o SOBP: prox R98 – dist R90
#       o DFO
#       o Uniforminty: (Max-Min)/Max
#•      load and compare reference values
#•      plot the dose and depth in a diagram


        gui.HideAll()
        gui.groupBox_DD_BluePhantom.setHidden(False)
        gui.textEdit_MessageField.setText("")
        print "WaterPhantomDepthDose"
        gui.labelAnalyseMenu.setText("Depth Dose Analysing - Water Phantom")

        inputFile=gui.FileToOpen()
        option=os.path.basename(inputFile)[0:8]                 # at the beginning of the
filename stands the range and modulation


        x_arr, y_arr, z_arr, d_arr=gui.LoadDatafromOPA(inputFile,1)     # 1 stands for first
measurment series
        if z_arr[0]>z_arr[-1]:
            x_arr=x_arr[::-1]
            y_arr=y_arr[::-1]
            z_arr=z_arr[::-1]
            d_arr=d_arr[::-1]
```

```
        d_norm=gui.NormalizeMax100(d_arr)

        gui.CalculateDDBPVal(d_norm,z_arr)
        gui.LoadDDBPIV(option, "Depth Dose - Water Phantom.txt")

        gui.ShowProfil(0,d_norm, 0, 105, "rel. Dose [%]",
                    z_arr, min(z_arr), max(z_arr),
                    "[mm]", "normalized Depth Dose")


    def WaterPhantomLateralDose(self, checked=None):                    # in Progress
#•      set the group boxes for lateral dose measurements with water phantom (blue
phantom) visible
#•      set text of heading
#•      ask for file to open
#•      extract option (range & modulation) from the input filename (first 8 letter)
#•      load coordinates and doses of the first and second measurement series
#•      identify measurement series to inline and crossline
#•      normalize the dose to maximum
#•      calculate all values that belong to the measurement
#       o field size
#       o penumbra low
#       o penumbra high
#       o lateral offset
#       o uniformity region
#       o uniformity: (Max-Min)/Max
#•      load and compare reference values
#•      plot the dose of inline and crossline in a diagram

        gui.HideAll()
        gui.groupBox_LateralDose_crossline.setHidden(False)
        gui.groupBox_LateralDose_inline.setHidden(False)
        gui.textEdit_MessageField.setText("")
        print "WaterPhantomLateralDose"
        gui.labelAnalyseMenu.setText("Lateral Dose Analysing - Water Phantom")
        k=0

        inputFile=gui.FileToOpen()
        option=os.path.basename(inputFile)[0:8]

        x_arr, y_arr, z_arr, d_arr=gui.LoadDatafromOPA(inputFile,1)
        i=gui.Inline(x_arr,y_arr)
        if i==1:
            print "InLine"
            k=k+3
            x_il=1*x_arr
            y_il=1*y_arr
            z_il=1*z_arr
            d_il=gui.NormalizeMax100(d_arr)
        elif i==2:
            print "CrossLine"
            k=k+2
            x_cl=1*x_arr
            y_cl=1*y_arr
```

```
    z_cl=1*z_arr
    d_cl=gui.NormalizeMax100(d_arr)
else:
    gui.textEdit_MessageField.setText("Coordinates drift too much!")
    k=20

x_arr, y_arr, z_arr, d_arr=gui.LoadDatafromOPA(inputFile,2)
i=gui.Inline(x_arr,y_arr)

if i==1:
    print "InLine"
    k=k-2
    x_il=1*x_arr
    y_il=1*y_arr
    z_il=1*z_arr
    d_il=gui.NormalizeMax100(d_arr)
elif i==2:
    print "CrossLine"
    k=k-3
    x_cl=1*x_arr
    y_cl=1*y_arr
    z_cl=1*z_arr
    d_cl=gui.NormalizeMax100(d_arr)
else:
    k=20
    gui.textEdit_MessageField.setText("Coordinates drift too much!")

if abs(k)==1:
    gui.textEdit_MessageField.setText("Measurment series both times"+\
                                      " at the same line "+\
                                      "(inline or crossline).")
if k>10:
    gui.textEdit_MessageField.setText("one or more measurment"+\
                                      " series drift too much.")

if x_il[0]>x_il[-1]:
    x_il=x_il[::-1]
    d_il=d_il[::-1]
if y_cl[0]>y_cl[-1]:
    y_cl=y_cl[::-1]
    d_cl=d_cl[::-1]

gui.label_LDBP_IL_xCoord_val.setText("["+str(x_il[0])+";"+\
                                     str(x_il[-1])+"]")
gui.label_LDBP_IL_yCoord_val.setText(str(y_il[int(len(y_il)/2)]))
gui.label_LDBP_IL_zCoord_val.setText(str(z_il[int(len(z_il)/2)]))

gui.label_LDBP_CL_xCoord_val.setText(str(x_cl[int(len(x_cl)/2)]))
gui.label_LDBP_CL_yCoord_val.setText("["+str(y_cl[0])+";"+\
                                     str(y_cl[-1])+"]")
gui.label_LDBP_CL_zCoord_val.setText(str(z_cl[int(len(z_cl)/2)]))

gui.CalculateLDBPVal(x_il,d_il,y_cl,d_cl)
gui.LoadLDBPIV(option, "Lateral Dose - Water Phantom.txt")
```

```
        gui.ShowProfil(0,d_il, 0, 105, "rel. Dose [%]",
                    x_il, min(x_il), max(x_il), "[mm]", "Inline")
        gui.ShowProfil(0,d_cl, 0, 105, "rel. Dose [%]",
                    y_cl, min(y_cl), max(y_cl), "[mm]", "Crossline")


    def DailyQADoubleScatteringDepthDoseLYNX(self, checked=None):          #
ready
#•      set the group box for depth dose measurements with the water phantom (blue
phantom) visible
#•      set text of heading
#•      ask for file to open
#•      extract option from the input filename (3rd-5th letter)
#•      load coordinates and doses of the first measurement series
#•      normalize the doses to the maximum
#•      calculate all values that belong to the measurement
#       o R1-Scan
#       o R2-Scan
#       o P1-Scan
#       o P2-Scan
#       o P3-Scan
#       o P4-Scan
#•      Load and compare reference values
#•      Plot the dose along the scans in two diagrams
#•      Normalize the doses to a greyscale
#•      Convert the dose values to an image
#•      Show the image

        gui.HideAll()
        gui.groupBox_DD_WedgePhantom.setHidden(False)
        gui.textEdit_MessageField.setText("")
        print "DailyQADoubleScatteringDepthDoseLYNX"
        gui.labelAnalyseMenu.setText("Daily QA - Double Scattering"+\
                            " - Depth Dose - LYNX (Wedge Phantom)")

        inputFile=gui.FileToOpen()
        option=os.path.basename(inputFile)[2:5]
        try:
            data_array,array_x,array_y= gui.LoadFromLYNXFile(inputFile)
        except ValueError:
            gui.textEdit_MessageField.setText(\
                "ERROR \n" \
                "Can not load Data from the selected File. \n" \
                "Please try again.")

        array_norm=gui.NormalizeMax100(data_array)

        gui.CalculateDDWPVal(array_norm,array_x,array_y,option)
        gui.LoadDQADSDDLYNXIV(option, "Depth Dose - Wedge Phantom - LYNX.txt")

        gui.ShowRScans(array_norm, array_x, array_y)
        gui.ShowPScans(array_norm, array_x, array_y, option)

        array_image=gui.GrayScale(data_array)
```

```
        img_path=gui.CreateImage(array_image)
        gui.ImageShow(img_path)


    def DailyQADoubleScatteringLateralDoseLYNX(self, checked=None):          # ready
#•      set the group box for depth dose measurements with the wedged phantom visi-
ble
#•      set text of heading
#•      ask for file to open
#•      extract option from the input filename (3rd-5th letter)
#•      load coordinates and dose equivalent values of the measurement series
#•      normalize the dose to maximum
#•      extract the values for inline and crossline
#•      calculate all values that belong to the measurement
#       o field size
#       o penumbra low
#       o penumbra high
#       o lateral offset
#       o uniformity region
#       o uniformity: (Max-Min)/Max
#•      load and compare reference values
#•      plot the dose of inline and crossline in a diagram

        gui.HideAll()
        print "DailyQADoubleScatteringLateralDoseLYNX"
        gui.labelAnalyseMenu.setText("Daily QA - Double Scattering"+\
                                " - Lateral Dose - LYNX (Flat Phantom)")
        gui.groupBox_LateralDose_inline.setHidden(False)
        gui.groupBox_LateralDose_crossline.setHidden(False)
        gui.textEdit_MessageField.setText("")
        inputfile=gui.FileToOpen()
        option=os.path.basename(inputfile)[2:5]
        try:
            data_array,x_il,y_cl= gui.LoadFromLYNXFile(inputfile)
        except ValueError:
            gui.textEdit_MessageField.setText(\
                  "ERROR \n" \
                  "Can not load Data from the selected File. \n" \
                  "Please try again.")
        array_norm=gui.NormalizeMax100(data_array)
        d_il=gui.ExtractArray(array_norm,int(len(y_cl)/2),int(len(y_cl)/2),0,len(x_il))
        d_cl=gui.ExtractArray(array_norm,0,len(y_cl),int(len(x_il)/2),int(len(x_il)/2))

        gui.label_LDBP_IL_xCoord_val.setText("["+str(x_il[0])+";"+\
                                    str(x_il[-1])+"]")
        gui.label_LDBP_IL_yCoord_val.setText(str(y_cl[int(len(y_cl)/2)]))
        gui.label_LDBP_IL_zCoord_val.setText("0")

        gui.label_LDBP_CL_xCoord_val.setText(str(x_il[int(len(x_il)/2)]))
        gui.label_LDBP_CL_yCoord_val.setText("["+str(y_cl[0])+";"+\
                                    str(y_cl[-1])+"]")
        gui.label_LDBP_CL_zCoord_val.setText("0")

        gui.CalculateLDBPVal(x_il,d_il,y_cl,d_cl)
        gui.LoadLDBPIV(option, "Lateral Dose - Water Phantom.txt")
```

```
        gui.ShowProfil(0,d_il, 0, 105, "rel. Dose [%]",
                    x_il, min(x_il), max(x_il), "[mm]", "Inline")
        gui.ShowProfil(0,d_cl, 0, 105, "rel. Dose [%]",
                    y_cl, min(y_cl), max(y_cl), "[mm]", "Crossline")

# Hauptprogramm
if __name__ == "__main__":
    # Applikationsobjekt erzeugen
    app = QtGui.QApplication(sys.argv)
    # Instanz anlegen
    gui = MainWindowGui()

    # Widget anzeigen
    gui.show()

    gui.HideAll()

#    gui.pushButtonRefresh.clicked.connect(gui.Refresh)
    gui.pushButton_Export.clicked.connect(gui.ExportPdf)
    gui.pushButton_DDWP_LoadOption.clicked.connect(gui.LoadOption)
    gui.pushButton_DDBP_LoadOption.clicked.connect(gui.LoadOption)
    gui.pushButton_LDBP_LoadOption.clicked.connect(gui.LoadOption)
    gui.actionDepth_Dose_LYNX.triggered.connect(\
                            gui.DailyQADoubleScatteringDepthDoseLYNX)
    gui.actionLateral_Profile_LYNX.triggered.connect(\
                            gui.DailyQADoubleScatteringLateralDoseLYNX)
    gui.actionDepth_Dose_WaterPhantom.triggered.connect(\
                            gui.WaterPhantomDepthDose)
    gui.actionLateral_Dose_WaterPhantom.triggered.connect(\
                            gui.WaterPhantomLateralDose)

    # Hauptschleife
    sys.exit(app.exec_())
```

Anhang A.2 – BaQtDesigner

```
# -*- coding: utf-8 -*-

# Form implementation generated from reading ui file 'BaQtDesigner.ui'
#
# Created: Thu Jul 10 11:02:53 2014
#      by: PyQt4 UI code generator 4.9.6
#
# WARNING! All changes made in this file will be lost!

from PyQt4 import QtCore, QtGui

try:
    _fromUtf8 = QtCore.QString.fromUtf8
except AttributeError:
    def _fromUtf8(s):
        return s

try:
    _encoding = QtGui.QApplication.UnicodeUTF8
    def _translate(context, text, disambig):
        return QtGui.QApplication.translate(context, text, disambig, _encoding)
except AttributeError:
    def _translate(context, text, disambig):
        return QtGui.QApplication.translate(context, text, disambig)

class Ui_MainWindow(object):
    def setupUi(self, MainWindow):
        MainWindow.setObjectName(_fromUtf8("MainWindow"))
        MainWindow.resize(962, 746)
        self.centralwidget = QtGui.QWidget(MainWindow)
        self.centralwidget.setObjectName(_fromUtf8("centralwidget"))
        self.widget_1 = MatplotLibWidget(self.centralwidget)
        self.widget_1.setGeometry(QtCore.QRect(440, 40, 501, 321))
        self.widget_1.setObjectName(_fromUtf8("widget_1"))
        self.lineEdit_FilePath = QtGui.QLineEdit(self.centralwidget)
        self.lineEdit_FilePath.setGeometry(QtCore.QRect(10, 50, 291, 21))
        self.lineEdit_FilePath.setReadOnly(True)
        self.lineEdit_FilePath.setObjectName(_fromUtf8("lineEdit_FilePath"))
        self.groupBox_LateralDose_inline = QtGui.QGroupBox(self.centralwidget)
        self.groupBox_LateralDose_inline.setGeometry(QtCore.QRect(10, 80, 421, 251))

self.groupBox_LateralDose_inline.setObjectName(_fromUtf8("groupBox_LateralDose_i
nline"))
        self.label_LDBP_IL_FieldSize = QtGui.QLabel(self.groupBox_LateralDose_inline)
        self.label_LDBP_IL_FieldSize.setGeometry(QtCore.QRect(20, 90, 46, 21))

self.label_LDBP_IL_FieldSize.setObjectName(_fromUtf8("label_LDBP_IL_FieldSize"))
        self.label_LDBP_IL_Pen_low = QtGui.QLabel(self.groupBox_LateralDose_inline)
        self.label_LDBP_IL_Pen_low.setGeometry(QtCore.QRect(20, 120, 81, 21))

self.label_LDBP_IL_Pen_low.setObjectName(_fromUtf8("label_LDBP_IL_Pen_low"))
        self.label_LDBP_IL_Pen_high = QtGui.QLabel(self.groupBox_LateralDose_inline)
```

```
        self.label_LDBP_IL_Pen_high.setGeometry(QtCore.QRect(20, 140, 91, 21))

self.label_LDBP_IL_Pen_high.setObjectName(_fromUtf8("label_LDBP_IL_Pen_high"))
        self.label_LDBP_IL_LatOff = QtGui.QLabel(self.groupBox_LateralDose_inline)
        self.label_LDBP_IL_LatOff.setGeometry(QtCore.QRect(20, 170, 81, 21))
        self.label_LDBP_IL_LatOff.setObjectName(_fromUtf8("label_LDBP_IL_LatOff"))
        self.label_LDBP_IL_UnifReg = QtGui.QLabel(self.groupBox_LateralDose_inline)
        self.label_LDBP_IL_UnifReg.setGeometry(QtCore.QRect(20, 200, 91, 21))

self.label_LDBP_IL_UnifReg.setObjectName(_fromUtf8("label_LDBP_IL_UnifReg"))
        self.label_LDBP_IL_MinMax = QtGui.QLabel(self.groupBox_LateralDose_inline)
        self.label_LDBP_IL_MinMax.setGeometry(QtCore.QRect(20, 220, 101, 21))

self.label_LDBP_IL_MinMax.setObjectName(_fromUtf8("label_LDBP_IL_MinMax"))
        self.label_LDBP_IL_Pen_low_rval =
QtGui.QLabel(self.groupBox_LateralDose_inline)
        self.label_LDBP_IL_Pen_low_rval.setGeometry(QtCore.QRect(130, 120, 41, 21))

self.label_LDBP_IL_Pen_low_rval.setObjectName(_fromUtf8("label_LDBP_IL_Pen_lo
w_rval"))
        self.label_LDBP_IL_UnifReg_rval =
QtGui.QLabel(self.groupBox_LateralDose_inline)
        self.label_LDBP_IL_UnifReg_rval.setGeometry(QtCore.QRect(130, 200, 71, 21))

self.label_LDBP_IL_UnifReg_rval.setObjectName(_fromUtf8("label_LDBP_IL_UnifReg
_rval"))
        self.label_LDBP_IL_FieldSize_rval =
QtGui.QLabel(self.groupBox_LateralDose_inline)
        self.label_LDBP_IL_FieldSize_rval.setGeometry(QtCore.QRect(130, 90, 41, 21))

self.label_LDBP_IL_FieldSize_rval.setObjectName(_fromUtf8("label_LDBP_IL_FieldSi
ze_rval"))
        self.label_LDBP_IL_LatOff_rval =
QtGui.QLabel(self.groupBox_LateralDose_inline)
        self.label_LDBP_IL_LatOff_rval.setGeometry(QtCore.QRect(130, 170, 41, 21))

self.label_LDBP_IL_LatOff_rval.setObjectName(_fromUtf8("label_LDBP_IL_LatOff_rva
l"))
        self.label_LDBP_IL_MinMax_rval =
QtGui.QLabel(self.groupBox_LateralDose_inline)
        self.label_LDBP_IL_MinMax_rval.setGeometry(QtCore.QRect(130, 220, 41, 21))

self.label_LDBP_IL_MinMax_rval.setObjectName(_fromUtf8("label_LDBP_IL_MinMax_
rval"))
        self.label_LDBP_IL_Pen_high_rval =
QtGui.QLabel(self.groupBox_LateralDose_inline)
        self.label_LDBP_IL_Pen_high_rval.setGeometry(QtCore.QRect(130, 140, 41, 21))

self.label_LDBP_IL_Pen_high_rval.setObjectName(_fromUtf8("label_LDBP_IL_Pen_hi
gh_rval"))
        self.label_LDBP_IL_Pen_high_ival =
QtGui.QLabel(self.groupBox_LateralDose_inline)
        self.label_LDBP_IL_Pen_high_ival.setGeometry(QtCore.QRect(200, 140, 41, 21))
```

```
self.label_LDBP_IL_Pen_high_ival.setObjectName(_fromUtf8("label_LDBP_IL_Pen_high_ival"))
        self.label_LDBP_IL_Pen_low_ival = QtGui.QLabel(self.groupBox_LateralDose_inline)
        self.label_LDBP_IL_Pen_low_ival.setGeometry(QtCore.QRect(200, 120, 41, 21))

self.label_LDBP_IL_Pen_low_ival.setObjectName(_fromUtf8("label_LDBP_IL_Pen_low_ival"))
        self.label_LDBP_IL_MinMax_ival = QtGui.QLabel(self.groupBox_LateralDose_inline)
        self.label_LDBP_IL_MinMax_ival.setGeometry(QtCore.QRect(200, 220, 41, 21))

self.label_LDBP_IL_MinMax_ival.setObjectName(_fromUtf8("label_LDBP_IL_MinMax_ival"))
        self.label_LDBP_IL_FieldSize_ival = QtGui.QLabel(self.groupBox_LateralDose_inline)
        self.label_LDBP_IL_FieldSize_ival.setGeometry(QtCore.QRect(200, 90, 41, 21))

self.label_LDBP_IL_FieldSize_ival.setObjectName(_fromUtf8("label_LDBP_IL_FieldSize_ival"))
        self.label_LDBP_IL_LatOff_ival = QtGui.QLabel(self.groupBox_LateralDose_inline)
        self.label_LDBP_IL_LatOff_ival.setGeometry(QtCore.QRect(200, 170, 41, 21))

self.label_LDBP_IL_LatOff_ival.setObjectName(_fromUtf8("label_LDBP_IL_LatOff_ival"))
        self.label_LDBP_IL_Pen_low_tol = QtGui.QLabel(self.groupBox_LateralDose_inline)
        self.label_LDBP_IL_Pen_low_tol.setGeometry(QtCore.QRect(280, 120, 41, 21))

self.label_LDBP_IL_Pen_low_tol.setObjectName(_fromUtf8("label_LDBP_IL_Pen_low_tol"))
        self.label_LDBP_IL_FieldSize_tol = QtGui.QLabel(self.groupBox_LateralDose_inline)
        self.label_LDBP_IL_FieldSize_tol.setGeometry(QtCore.QRect(280, 90, 41, 21))

self.label_LDBP_IL_FieldSize_tol.setObjectName(_fromUtf8("label_LDBP_IL_FieldSize_tol"))
        self.label_LDBP_IL_LatOff_tol = QtGui.QLabel(self.groupBox_LateralDose_inline)
        self.label_LDBP_IL_LatOff_tol.setGeometry(QtCore.QRect(280, 170, 41, 21))

self.label_LDBP_IL_LatOff_tol.setObjectName(_fromUtf8("label_LDBP_IL_LatOff_tol"))
        self.label_LDBP_IL_Pen_high_tol = QtGui.QLabel(self.groupBox_LateralDose_inline)
        self.label_LDBP_IL_Pen_high_tol.setGeometry(QtCore.QRect(280, 140, 41, 21))

self.label_LDBP_IL_Pen_high_tol.setObjectName(_fromUtf8("label_LDBP_IL_Pen_high_tol"))
        self.label_LDBP_IL_MinMax_tol = QtGui.QLabel(self.groupBox_LateralDose_inline)
        self.label_LDBP_IL_MinMax_tol.setGeometry(QtCore.QRect(280, 220, 41, 21))

self.label_LDBP_IL_MinMax_tol.setObjectName(_fromUtf8("label_LDBP_IL_MinMax_tol"))
```

```
        self.label_LDBP_IL_relValue = QtGui.QLabel(self.groupBox_LateralDose_inline)
        self.label_LDBP_IL_relValue.setGeometry(QtCore.QRect(130, 60, 61, 21))
        font = QtGui.QFont()
        font.setBold(True)
        font.setWeight(75)
        self.label_LDBP_IL_relValue.setFont(font)

self.label_LDBP_IL_relValue.setObjectName(_fromUtf8("label_LDBP_IL_relValue"))
        self.label_LDBP_IL_tolerance = QtGui.QLabel(self.groupBox_LateralDose_inline)
        self.label_LDBP_IL_tolerance.setGeometry(QtCore.QRect(280, 60, 61, 21))
        font = QtGui.QFont()
        font.setBold(True)
        font.setWeight(75)
        self.label_LDBP_IL_tolerance.setFont(font)

self.label_LDBP_IL_tolerance.setObjectName(_fromUtf8("label_LDBP_IL_tolerance"))
        self.label_LDBP_IL_idealValue =
QtGui.QLabel(self.groupBox_LateralDose_inline)
        self.label_LDBP_IL_idealValue.setGeometry(QtCore.QRect(200, 60, 71, 21))
        font = QtGui.QFont()
        font.setBold(True)
        font.setWeight(75)
        self.label_LDBP_IL_idealValue.setFont(font)

self.label_LDBP_IL_idealValue.setObjectName(_fromUtf8("label_LDBP_IL_idealValue"
))
        self.label_LDBP_IL_xCoord = QtGui.QLabel(self.groupBox_LateralDose_inline)
        self.label_LDBP_IL_xCoord.setGeometry(QtCore.QRect(20, 20, 16, 21))
        self.label_LDBP_IL_xCoord.setObjectName(_fromUtf8("label_LDBP_IL_xCoord"))
        self.label_LDBP_IL_yCoord = QtGui.QLabel(self.groupBox_LateralDose_inline)
        self.label_LDBP_IL_yCoord.setGeometry(QtCore.QRect(150, 20, 16, 21))
        self.label_LDBP_IL_yCoord.setObjectName(_fromUtf8("label_LDBP_IL_yCoord"))
        self.label_LDBP_IL_zCoord = QtGui.QLabel(self.groupBox_LateralDose_inline)
        self.label_LDBP_IL_zCoord.setGeometry(QtCore.QRect(300, 20, 16, 21))
        self.label_LDBP_IL_zCoord.setObjectName(_fromUtf8("label_LDBP_IL_zCoord"))
        self.label_LDBP_IL_xCoord_val =
QtGui.QLabel(self.groupBox_LateralDose_inline)
        self.label_LDBP_IL_xCoord_val.setGeometry(QtCore.QRect(40, 20, 81, 21))

self.label_LDBP_IL_xCoord_val.setObjectName(_fromUtf8("label_LDBP_IL_xCoord_v
al"))
        self.label_LDBP_IL_yCoord_val =
QtGui.QLabel(self.groupBox_LateralDose_inline)
        self.label_LDBP_IL_yCoord_val.setGeometry(QtCore.QRect(170, 20, 81, 21))

self.label_LDBP_IL_yCoord_val.setObjectName(_fromUtf8("label_LDBP_IL_yCoord_v
al"))
        self.label_LDBP_IL_zCoord_val =
QtGui.QLabel(self.groupBox_LateralDose_inline)
        self.label_LDBP_IL_zCoord_val.setGeometry(QtCore.QRect(320, 20, 81, 21))

self.label_LDBP_IL_zCoord_val.setObjectName(_fromUtf8("label_LDBP_IL_zCoord_v
al"))
        self.label_LDBP_IL_Deviation = QtGui.QLabel(self.groupBox_LateralDose_inline)
        self.label_LDBP_IL_Deviation.setGeometry(QtCore.QRect(350, 60, 61, 21))
```

```
        font = QtGui.QFont()
        font.setBold(True)
        font.setWeight(75)
        self.label_LDBP_IL_Deviation.setFont(font)

self.label_LDBP_IL_Deviation.setObjectName(_fromUtf8("label_LDBP_IL_Deviation"))
        self.label_LDBP_IL_Pen_high_dev =
QtGui.QLabel(self.groupBox_LateralDose_inline)
        self.label_LDBP_IL_Pen_high_dev.setGeometry(QtCore.QRect(350, 140, 41, 21))

self.label_LDBP_IL_Pen_high_dev.setObjectName(_fromUtf8("label_LDBP_IL_Pen_hi
gh_dev"))
        self.label_LDBP_IL_FieldSize_dev =
QtGui.QLabel(self.groupBox_LateralDose_inline)
        self.label_LDBP_IL_FieldSize_dev.setGeometry(QtCore.QRect(350, 90, 41, 21))

self.label_LDBP_IL_FieldSize_dev.setObjectName(_fromUtf8("label_LDBP_IL_FieldSi
ze_dev"))
        self.label_LDBP_IL_Pen_low_dev =
QtGui.QLabel(self.groupBox_LateralDose_inline)
        self.label_LDBP_IL_Pen_low_dev.setGeometry(QtCore.QRect(350, 120, 41, 21))

self.label_LDBP_IL_Pen_low_dev.setObjectName(_fromUtf8("label_LDBP_IL_Pen_lo
w_dev"))
        self.label_LDBP_IL_MinMax_dev =
QtGui.QLabel(self.groupBox_LateralDose_inline)
        self.label_LDBP_IL_MinMax_dev.setGeometry(QtCore.QRect(350, 220, 41, 21))

self.label_LDBP_IL_MinMax_dev.setObjectName(_fromUtf8("label_LDBP_IL_MinMax_
dev"))
        self.label_LDBP_IL_LatOff_dev =
QtGui.QLabel(self.groupBox_LateralDose_inline)
        self.label_LDBP_IL_LatOff_dev.setGeometry(QtCore.QRect(350, 170, 41, 21))

self.label_LDBP_IL_LatOff_dev.setObjectName(_fromUtf8("label_LDBP_IL_LatOff_de
v"))
        self.pushButton_LDBP_LoadOption =
QtGui.QPushButton(self.groupBox_LateralDose_inline)
        self.pushButton_LDBP_LoadOption.setGeometry(QtCore.QRect(260, 40, 75, 23))

self.pushButton_LDBP_LoadOption.setObjectName(_fromUtf8("pushButton_LDBP_Lo
adOption"))
        self.lineEdit_LDBP_option = QtGui.QLineEdit(self.groupBox_LateralDose_inline)
        self.lineEdit_LDBP_option.setGeometry(QtCore.QRect(190, 40, 61, 21))
        self.lineEdit_LDBP_option.setText(_fromUtf8(""))
        self.lineEdit_LDBP_option.setObjectName(_fromUtf8("lineEdit_LDBP_option"))
        self.label_LDBP_option = QtGui.QLabel(self.groupBox_LateralDose_inline)
        self.label_LDBP_option.setGeometry(QtCore.QRect(130, 40, 61, 21))
        self.label_LDBP_option.setObjectName(_fromUtf8("label_LDBP_option"))
        self.label_LDBP_RangeModulation =
QtGui.QLabel(self.groupBox_LateralDose_inline)
        self.label_LDBP_RangeModulation.setGeometry(QtCore.QRect(20, 40, 101, 21))

self.label_LDBP_RangeModulation.setObjectName(_fromUtf8("label_LDBP_RangeMo
dulation"))
```

```
        self.groupBox_DD_BluePhantom = QtGui.QGroupBox(self.centralwidget)
        self.groupBox_DD_BluePhantom.setGeometry(QtCore.QRect(10, 80, 421, 401))

self.groupBox_DD_BluePhantom.setObjectName(_fromUtf8("groupBox_DD_BluePhan
tom"))
        self.label_DDBP_dR80_rval = QtGui.QLabel(self.groupBox_DD_BluePhantom)
        self.label_DDBP_dR80_rval.setGeometry(QtCore.QRect(130, 210, 41, 21))

self.label_DDBP_dR80_rval.setObjectName(_fromUtf8("label_DDBP_dR80_rval"))
        self.label_DDBP_dR90 = QtGui.QLabel(self.groupBox_DD_BluePhantom)
        self.label_DDBP_dR90.setGeometry(QtCore.QRect(10, 230, 61, 21))
        self.label_DDBP_dR90.setObjectName(_fromUtf8("label_DDBP_dR90"))
        self.label_DDBP_dR20 = QtGui.QLabel(self.groupBox_DD_BluePhantom)
        self.label_DDBP_dR20.setGeometry(QtCore.QRect(10, 170, 61, 21))
        self.label_DDBP_dR20.setObjectName(_fromUtf8("label_DDBP_dR20"))
        self.label_DDBP_dR90_rval = QtGui.QLabel(self.groupBox_DD_BluePhantom)
        self.label_DDBP_dR90_rval.setGeometry(QtCore.QRect(130, 230, 41, 21))

self.label_DDBP_dR90_rval.setObjectName(_fromUtf8("label_DDBP_dR90_rval"))
        self.label_DDBP_dR80 = QtGui.QLabel(self.groupBox_DD_BluePhantom)
        self.label_DDBP_dR80.setGeometry(QtCore.QRect(10, 210, 61, 21))
        self.label_DDBP_dR80.setObjectName(_fromUtf8("label_DDBP_dR80"))
        self.label_DDBP_dR20_rval = QtGui.QLabel(self.groupBox_DD_BluePhantom)
        self.label_DDBP_dR20_rval.setGeometry(QtCore.QRect(130, 170, 41, 21))

self.label_DDBP_dR20_rval.setObjectName(_fromUtf8("label_DDBP_dR20_rval"))
        self.label_DDBP_SOBP_9090 = QtGui.QLabel(self.groupBox_DD_BluePhantom)
        self.label_DDBP_SOBP_9090.setGeometry(QtCore.QRect(10, 280, 111, 21))

self.label_DDBP_SOBP_9090.setObjectName(_fromUtf8("label_DDBP_SOBP_9090"))
        self.label_DDBP_DFO_rval = QtGui.QLabel(self.groupBox_DD_BluePhantom)
        self.label_DDBP_DFO_rval.setGeometry(QtCore.QRect(130, 330, 41, 21))
        self.label_DDBP_DFO_rval.setObjectName(_fromUtf8("label_DDBP_DFO_rval"))
        self.label_DDBP_SOBP_9090_rval =
QtGui.QLabel(self.groupBox_DD_BluePhantom)
        self.label_DDBP_SOBP_9090_rval.setGeometry(QtCore.QRect(130, 280, 41,
21))

self.label_DDBP_SOBP_9090_rval.setObjectName(_fromUtf8("label_DDBP_SOBP_90
90_rval"))
        self.label_DDBP_DFO = QtGui.QLabel(self.groupBox_DD_BluePhantom)
        self.label_DDBP_DFO.setGeometry(QtCore.QRect(10, 330, 111, 21))
        self.label_DDBP_DFO.setObjectName(_fromUtf8("label_DDBP_DFO"))
        self.label_DDBP_dR80_ival = QtGui.QLabel(self.groupBox_DD_BluePhantom)
        self.label_DDBP_dR80_ival.setGeometry(QtCore.QRect(200, 210, 41, 21))
        self.label_DDBP_dR80_ival.setObjectName(_fromUtf8("label_DDBP_dR80_ival"))
        self.label_DDBP_DFO_ival = QtGui.QLabel(self.groupBox_DD_BluePhantom)
        self.label_DDBP_DFO_ival.setGeometry(QtCore.QRect(200, 330, 41, 21))
        self.label_DDBP_DFO_ival.setObjectName(_fromUtf8("label_DDBP_DFO_ival"))
        self.label_DDBP_dR90_ival = QtGui.QLabel(self.groupBox_DD_BluePhantom)
        self.label_DDBP_dR90_ival.setGeometry(QtCore.QRect(200, 230, 41, 21))
        self.label_DDBP_dR90_ival.setObjectName(_fromUtf8("label_DDBP_dR90_ival"))
        self.label_DDBP_dR20_ival = QtGui.QLabel(self.groupBox_DD_BluePhantom)
        self.label_DDBP_dR20_ival.setGeometry(QtCore.QRect(200, 170, 41, 21))
        self.label_DDBP_dR20_ival.setObjectName(_fromUtf8("label_DDBP_dR20_ival"))
```

```
        self.label_DDBP_SOBP_9090_ival =
QtGui.QLabel(self.groupBox_DD_BluePhantom)
        self.label_DDBP_SOBP_9090_ival.setGeometry(QtCore.QRect(200, 280, 41, 21))

self.label_DDBP_SOBP_9090_ival.setObjectName(_fromUtf8("label_DDBP_SOBP_90
90_ival"))
        self.label_DDBP_pR98_rval = QtGui.QLabel(self.groupBox_DD_BluePhantom)
        self.label_DDBP_pR98_rval.setGeometry(QtCore.QRect(130, 120, 41, 21))

self.label_DDBP_pR98_rval.setObjectName(_fromUtf8("label_DDBP_pR98_rval"))
        self.label_DDBP_pR95_ival = QtGui.QLabel(self.groupBox_DD_BluePhantom)
        self.label_DDBP_pR95_ival.setGeometry(QtCore.QRect(200, 100, 41, 21))
        self.label_DDBP_pR95_ival.setObjectName(_fromUtf8("label_DDBP_pR95_ival"))
        self.label_DDBP_pR90_ival = QtGui.QLabel(self.groupBox_DD_BluePhantom)
        self.label_DDBP_pR90_ival.setGeometry(QtCore.QRect(200, 80, 41, 21))
        self.label_DDBP_pR90_ival.setObjectName(_fromUtf8("label_DDBP_pR90_ival"))
        self.label_DDBP_pR98 = QtGui.QLabel(self.groupBox_DD_BluePhantom)
        self.label_DDBP_pR98.setGeometry(QtCore.QRect(10, 120, 71, 21))
        self.label_DDBP_pR98.setObjectName(_fromUtf8("label_DDBP_pR98"))
        self.label_DDBP_pR95 = QtGui.QLabel(self.groupBox_DD_BluePhantom)
        self.label_DDBP_pR95.setGeometry(QtCore.QRect(10, 100, 71, 21))
        self.label_DDBP_pR95.setObjectName(_fromUtf8("label_DDBP_pR95"))
        self.label_DDBP_pR90 = QtGui.QLabel(self.groupBox_DD_BluePhantom)
        self.label_DDBP_pR90.setGeometry(QtCore.QRect(10, 80, 71, 21))
        self.label_DDBP_pR90.setObjectName(_fromUtf8("label_DDBP_pR90"))
        self.label_DDBP_pR98_ival = QtGui.QLabel(self.groupBox_DD_BluePhantom)
        self.label_DDBP_pR98_ival.setGeometry(QtCore.QRect(200, 120, 41, 21))
        self.label_DDBP_pR98_ival.setObjectName(_fromUtf8("label_DDBP_pR98_ival"))
        self.label_DDBP_pR95_rval = QtGui.QLabel(self.groupBox_DD_BluePhantom)
        self.label_DDBP_pR95_rval.setGeometry(QtCore.QRect(130, 100, 41, 21))

self.label_DDBP_pR95_rval.setObjectName(_fromUtf8("label_DDBP_pR95_rval"))
        self.label_DDBP_pR90_rval = QtGui.QLabel(self.groupBox_DD_BluePhantom)
        self.label_DDBP_pR90_rval.setGeometry(QtCore.QRect(130, 80, 41, 21))

self.label_DDBP_pR90_rval.setObjectName(_fromUtf8("label_DDBP_pR90_rval"))
        self.label_DDBP_dR10 = QtGui.QLabel(self.groupBox_DD_BluePhantom)
        self.label_DDBP_dR10.setGeometry(QtCore.QRect(10, 150, 61, 21))
        self.label_DDBP_dR10.setObjectName(_fromUtf8("label_DDBP_dR10"))
        self.label_DDBP_dR10_ival = QtGui.QLabel(self.groupBox_DD_BluePhantom)
        self.label_DDBP_dR10_ival.setGeometry(QtCore.QRect(200, 150, 41, 21))
        self.label_DDBP_dR10_ival.setObjectName(_fromUtf8("label_DDBP_dR10_ival"))
        self.label_DDBP_dR10_rval = QtGui.QLabel(self.groupBox_DD_BluePhantom)
        self.label_DDBP_dR10_rval.setGeometry(QtCore.QRect(130, 150, 41, 21))

self.label_DDBP_dR10_rval.setObjectName(_fromUtf8("label_DDBP_dR10_rval"))
        self.label_DDBP_dR50 = QtGui.QLabel(self.groupBox_DD_BluePhantom)
        self.label_DDBP_dR50.setGeometry(QtCore.QRect(10, 190, 61, 21))
        self.label_DDBP_dR50.setObjectName(_fromUtf8("label_DDBP_dR50"))
        self.label_DDBP_dR50_ival = QtGui.QLabel(self.groupBox_DD_BluePhantom)
        self.label_DDBP_dR50_ival.setGeometry(QtCore.QRect(200, 190, 41, 21))
        self.label_DDBP_dR50_ival.setObjectName(_fromUtf8("label_DDBP_dR50_ival"))
        self.label_DDBP_dR50_rval = QtGui.QLabel(self.groupBox_DD_BluePhantom)
        self.label_DDBP_dR50_rval.setGeometry(QtCore.QRect(130, 190, 41, 21))
```

```
self.label_DDBP_dR50_rval.setObjectName(_fromUtf8("label_DDBP_dR50_rval"))
        self.label_DDBP_dR95 = QtGui.QLabel(self.groupBox_DD_BluePhantom)
        self.label_DDBP_dR95.setGeometry(QtCore.QRect(10, 250, 61, 21))
        self.label_DDBP_dR95.setObjectName(_fromUtf8("label_DDBP_dR95"))
        self.label_DDBP_dR95_ival = QtGui.QLabel(self.groupBox_DD_BluePhantom)
        self.label_DDBP_dR95_ival.setGeometry(QtCore.QRect(200, 250, 41, 21))
        self.label_DDBP_dR95_ival.setObjectName(_fromUtf8("label_DDBP_dR95_ival"))
        self.label_DDBP_dR95_rval = QtGui.QLabel(self.groupBox_DD_BluePhantom)
        self.label_DDBP_dR95_rval.setGeometry(QtCore.QRect(130, 250, 41, 21))

self.label_DDBP_dR95_rval.setObjectName(_fromUtf8("label_DDBP_dR95_rval"))
        self.label_DDBP_SOBP_9890_rval =
QtGui.QLabel(self.groupBox_DD_BluePhantom)
        self.label_DDBP_SOBP_9890_rval.setGeometry(QtCore.QRect(130, 300, 41,
21))

self.label_DDBP_SOBP_9890_rval.setObjectName(_fromUtf8("label_DDBP_SOBP_98
90_rval"))
        self.label_DDBP_SOBP_9890 = QtGui.QLabel(self.groupBox_DD_BluePhantom)
        self.label_DDBP_SOBP_9890.setGeometry(QtCore.QRect(10, 300, 111, 21))

self.label_DDBP_SOBP_9890.setObjectName(_fromUtf8("label_DDBP_SOBP_9890"))
        self.label_DDBP_SOBP_9890_ival =
QtGui.QLabel(self.groupBox_DD_BluePhantom)
        self.label_DDBP_SOBP_9890_ival.setGeometry(QtCore.QRect(200, 300, 41, 21))

self.label_DDBP_SOBP_9890_ival.setObjectName(_fromUtf8("label_DDBP_SOBP_98
90_ival"))
        self.label_DDBP_MinMax_ival = QtGui.QLabel(self.groupBox_DD_BluePhantom)
        self.label_DDBP_MinMax_ival.setGeometry(QtCore.QRect(200, 360, 41, 21))

self.label_DDBP_MinMax_ival.setObjectName(_fromUtf8("label_DDBP_MinMax_ival"))
        self.label_DDBP_MinMax = QtGui.QLabel(self.groupBox_DD_BluePhantom)
        self.label_DDBP_MinMax.setGeometry(QtCore.QRect(10, 360, 101, 21))
        self.label_DDBP_MinMax.setObjectName(_fromUtf8("label_DDBP_MinMax"))
        self.label_DDBP_MinMax_rval = QtGui.QLabel(self.groupBox_DD_BluePhantom)
        self.label_DDBP_MinMax_rval.setGeometry(QtCore.QRect(130, 360, 41, 21))

self.label_DDBP_MinMax_rval.setObjectName(_fromUtf8("label_DDBP_MinMax_rval")
)
        self.label_DDBP_MinMax_tol = QtGui.QLabel(self.groupBox_DD_BluePhantom)
        self.label_DDBP_MinMax_tol.setGeometry(QtCore.QRect(280, 360, 41, 21))

self.label_DDBP_MinMax_tol.setObjectName(_fromUtf8("label_DDBP_MinMax_tol"))
        self.label_DDBP_SOBP_9090_tol =
QtGui.QLabel(self.groupBox_DD_BluePhantom)
        self.label_DDBP_SOBP_9090_tol.setGeometry(QtCore.QRect(280, 280, 41, 21))

self.label_DDBP_SOBP_9090_tol.setObjectName(_fromUtf8("label_DDBP_SOBP_909
0_tol"))
        self.label_DDBP_pR95_tol = QtGui.QLabel(self.groupBox_DD_BluePhantom)
        self.label_DDBP_pR95_tol.setGeometry(QtCore.QRect(280, 100, 41, 21))
        self.label_DDBP_pR95_tol.setObjectName(_fromUtf8("label_DDBP_pR95_tol"))
        self.label_DDBP_DFO_tol = QtGui.QLabel(self.groupBox_DD_BluePhantom)
```

```
        self.label_DDBP_DFO_tol.setGeometry(QtCore.QRect(280, 330, 41, 21))
        self.label_DDBP_DFO_tol.setObjectName(_fromUtf8("label_DDBP_DFO_tol"))
        self.label_DDBP_dR95_tol = QtGui.QLabel(self.groupBox_DD_BluePhantom)
        self.label_DDBP_dR95_tol.setGeometry(QtCore.QRect(280, 250, 41, 21))
        self.label_DDBP_dR95_tol.setObjectName(_fromUtf8("label_DDBP_dR95_tol"))
        self.label_DDBP_dR20_tol = QtGui.QLabel(self.groupBox_DD_BluePhantom)
        self.label_DDBP_dR20_tol.setGeometry(QtCore.QRect(280, 170, 41, 21))
        self.label_DDBP_dR20_tol.setObjectName(_fromUtf8("label_DDBP_dR20_tol"))
        self.label_DDBP_SOBP_9890_tol =
QtGui.QLabel(self.groupBox_DD_BluePhantom)
        self.label_DDBP_SOBP_9890_tol.setGeometry(QtCore.QRect(280, 300, 41, 21))

self.label_DDBP_SOBP_9890_tol.setObjectName(_fromUtf8("label_DDBP_SOBP_989
0_tol"))
        self.label_DDBP_dR50_tol = QtGui.QLabel(self.groupBox_DD_BluePhantom)
        self.label_DDBP_dR50_tol.setGeometry(QtCore.QRect(280, 190, 41, 21))
        self.label_DDBP_dR50_tol.setObjectName(_fromUtf8("label_DDBP_dR50_tol"))
        self.label_DDBP_pR90_tol = QtGui.QLabel(self.groupBox_DD_BluePhantom)
        self.label_DDBP_pR90_tol.setGeometry(QtCore.QRect(280, 80, 41, 21))
        self.label_DDBP_pR90_tol.setObjectName(_fromUtf8("label_DDBP_pR90_tol"))
        self.label_DDBP_pR98_tol = QtGui.QLabel(self.groupBox_DD_BluePhantom)
        self.label_DDBP_pR98_tol.setGeometry(QtCore.QRect(280, 120, 41, 21))
        self.label_DDBP_pR98_tol.setObjectName(_fromUtf8("label_DDBP_pR98_tol"))
        self.label_DDBP_dR80_tol = QtGui.QLabel(self.groupBox_DD_BluePhantom)
        self.label_DDBP_dR80_tol.setGeometry(QtCore.QRect(280, 210, 41, 21))
        self.label_DDBP_dR80_tol.setObjectName(_fromUtf8("label_DDBP_dR80_tol"))
        self.label_DDBP_dR90_tol = QtGui.QLabel(self.groupBox_DD_BluePhantom)
        self.label_DDBP_dR90_tol.setGeometry(QtCore.QRect(280, 230, 41, 21))
        self.label_DDBP_dR90_tol.setObjectName(_fromUtf8("label_DDBP_dR90_tol"))
        self.label_DDBP_dR10_tol = QtGui.QLabel(self.groupBox_DD_BluePhantom)
        self.label_DDBP_dR10_tol.setGeometry(QtCore.QRect(280, 150, 41, 21))
        self.label_DDBP_dR10_tol.setObjectName(_fromUtf8("label_DDBP_dR10_tol"))
        self.label_DDBP_realValue = QtGui.QLabel(self.groupBox_DD_BluePhantom)
        self.label_DDBP_realValue.setGeometry(QtCore.QRect(130, 50, 61, 21))
        font = QtGui.QFont()
        font.setBold(True)
        font.setWeight(75)
        self.label_DDBP_realValue.setFont(font)
        self.label_DDBP_realValue.setObjectName(_fromUtf8("label_DDBP_realValue"))
        self.label_DDBP_idealValue = QtGui.QLabel(self.groupBox_DD_BluePhantom)
        self.label_DDBP_idealValue.setGeometry(QtCore.QRect(200, 50, 71, 21))
        font = QtGui.QFont()
        font.setBold(True)
        font.setWeight(75)
        self.label_DDBP_idealValue.setFont(font)

self.label_DDBP_idealValue.setObjectName(_fromUtf8("label_DDBP_idealValue"))
        self.label_DDBP_tolerance = QtGui.QLabel(self.groupBox_DD_BluePhantom)
        self.label_DDBP_tolerance.setGeometry(QtCore.QRect(280, 50, 61, 21))
        font = QtGui.QFont()
        font.setBold(True)
        font.setWeight(75)
        self.label_DDBP_tolerance.setFont(font)
        self.label_DDBP_tolerance.setObjectName(_fromUtf8("label_DDBP_tolerance"))
```

```
        self.label_DDBP_RangeModulation =
QtGui.QLabel(self.groupBox_DD_BluePhantom)
        self.label_DDBP_RangeModulation.setGeometry(QtCore.QRect(10, 20, 101, 21))

self.label_DDBP_RangeModulation.setObjectName(_fromUtf8("label_DDBP_RangeMo
dulation"))
        self.lineEdit_DDBP_option = QtGui.QLineEdit(self.groupBox_DD_BluePhantom)
        self.lineEdit_DDBP_option.setGeometry(QtCore.QRect(180, 20, 61, 21))
        self.lineEdit_DDBP_option.setText(_fromUtf8(""))
        self.lineEdit_DDBP_option.setObjectName(_fromUtf8("lineEdit_DDBP_option"))
        self.label_DDBP_option = QtGui.QLabel(self.groupBox_DD_BluePhantom)
        self.label_DDBP_option.setGeometry(QtCore.QRect(120, 20, 61, 21))
        self.label_DDBP_option.setObjectName(_fromUtf8("label_DDBP_option"))
        self.pushButton_DDBP_LoadOption =
QtGui.QPushButton(self.groupBox_DD_BluePhantom)
        self.pushButton_DDBP_LoadOption.setGeometry(QtCore.QRect(250, 20, 75, 23))

self.pushButton_DDBP_LoadOption.setObjectName(_fromUtf8("pushButton_DDBP_Lo
adOption"))
        self.label_DDBP_Deviation = QtGui.QLabel(self.groupBox_DD_BluePhantom)
        self.label_DDBP_Deviation.setGeometry(QtCore.QRect(350, 50, 61, 21))
        font = QtGui.QFont()
        font.setBold(True)
        font.setWeight(75)
        self.label_DDBP_Deviation.setFont(font)
        self.label_DDBP_Deviation.setObjectName(_fromUtf8("label_DDBP_Deviation"))
        self.label_DDBP_pR98_dev = QtGui.QLabel(self.groupBox_DD_BluePhantom)
        self.label_DDBP_pR98_dev.setGeometry(QtCore.QRect(350, 120, 41, 21))

self.label_DDBP_pR98_dev.setObjectName(_fromUtf8("label_DDBP_pR98_dev"))
        self.label_DDBP_pR95_dev = QtGui.QLabel(self.groupBox_DD_BluePhantom)
        self.label_DDBP_pR95_dev.setGeometry(QtCore.QRect(350, 100, 41, 21))

self.label_DDBP_pR95_dev.setObjectName(_fromUtf8("label_DDBP_pR95_dev"))
        self.label_DDBP_dR90_dev = QtGui.QLabel(self.groupBox_DD_BluePhantom)
        self.label_DDBP_dR90_dev.setGeometry(QtCore.QRect(350, 230, 41, 21))

self.label_DDBP_dR90_dev.setObjectName(_fromUtf8("label_DDBP_dR90_dev"))
        self.label_DDBP_dR95_dev = QtGui.QLabel(self.groupBox_DD_BluePhantom)
        self.label_DDBP_dR95_dev.setGeometry(QtCore.QRect(350, 250, 41, 21))

self.label_DDBP_dR95_dev.setObjectName(_fromUtf8("label_DDBP_dR95_dev"))
        self.label_DDBP_SOBP_9890_dev =
QtGui.QLabel(self.groupBox_DD_BluePhantom)
        self.label_DDBP_SOBP_9890_dev.setGeometry(QtCore.QRect(350, 300, 41,
21))

self.label_DDBP_SOBP_9890_dev.setObjectName(_fromUtf8("label_DDBP_SOBP_98
90_dev"))
        self.label_DDBP_pR90_dev = QtGui.QLabel(self.groupBox_DD_BluePhantom)
        self.label_DDBP_pR90_dev.setGeometry(QtCore.QRect(350, 80, 41, 21))

self.label_DDBP_pR90_dev.setObjectName(_fromUtf8("label_DDBP_pR90_dev"))
        self.label_DDBP_SOBP_9090_dev =
QtGui.QLabel(self.groupBox_DD_BluePhantom)
```

```
        self.label_DDBP_SOBP_9090_dev.setGeometry(QtCore.QRect(350, 280, 41,
21))

self.label_DDBP_SOBP_9090_dev.setObjectName(_fromUtf8("label_DDBP_SOBP_90
90_dev"))
        self.label_DDBP_dR80_dev = QtGui.QLabel(self.groupBox_DD_BluePhantom)
        self.label_DDBP_dR80_dev.setGeometry(QtCore.QRect(350, 210, 41, 21))

self.label_DDBP_dR80_dev.setObjectName(_fromUtf8("label_DDBP_dR80_dev"))
        self.label_DDBP_DFO_dev = QtGui.QLabel(self.groupBox_DD_BluePhantom)
        self.label_DDBP_DFO_dev.setGeometry(QtCore.QRect(350, 330, 41, 21))
        self.label_DDBP_DFO_dev.setObjectName(_fromUtf8("label_DDBP_DFO_dev"))
        self.label_DDBP_dR20_dev = QtGui.QLabel(self.groupBox_DD_BluePhantom)
        self.label_DDBP_dR20_dev.setGeometry(QtCore.QRect(350, 170, 41, 21))

self.label_DDBP_dR20_dev.setObjectName(_fromUtf8("label_DDBP_dR20_dev"))
        self.label_DDBP_dR10_dev = QtGui.QLabel(self.groupBox_DD_BluePhantom)
        self.label_DDBP_dR10_dev.setGeometry(QtCore.QRect(350, 150, 41, 21))

self.label_DDBP_dR10_dev.setObjectName(_fromUtf8("label_DDBP_dR10_dev"))
        self.label_DDBP_MinMax_dev = QtGui.QLabel(self.groupBox_DD_BluePhantom)
        self.label_DDBP_MinMax_dev.setGeometry(QtCore.QRect(350, 360, 41, 21))

self.label_DDBP_MinMax_dev.setObjectName(_fromUtf8("label_DDBP_MinMax_dev")
)
        self.label_DDBP_dR50_dev = QtGui.QLabel(self.groupBox_DD_BluePhantom)
        self.label_DDBP_dR50_dev.setGeometry(QtCore.QRect(350, 190, 41, 21))

self.label_DDBP_dR50_dev.setObjectName(_fromUtf8("label_DDBP_dR50_dev"))
        self.groupBox_LateralDose_crossline = QtGui.QGroupBox(self.centralwidget)
        self.groupBox_LateralDose_crossline.setGeometry(QtCore.QRect(10, 340, 421,
251))

self.groupBox_LateralDose_crossline.setObjectName(_fromUtf8("groupBox_LateralDo
se_crossline"))
        self.label_LDBP_CL_LatOff_rval =
QtGui.QLabel(self.groupBox_LateralDose_crossline)
        self.label_LDBP_CL_LatOff_rval.setGeometry(QtCore.QRect(130, 170, 41, 21))

self.label_LDBP_CL_LatOff_rval.setObjectName(_fromUtf8("label_LDBP_CL_LatOff_r
val"))
        self.label_LDBP_CL_UnifReg =
QtGui.QLabel(self.groupBox_LateralDose_crossline)
        self.label_LDBP_CL_UnifReg.setGeometry(QtCore.QRect(20, 200, 91, 21))

self.label_LDBP_CL_UnifReg.setObjectName(_fromUtf8("label_LDBP_CL_UnifReg"))
        self.label_LDBP_CL_MinMax_rval =
QtGui.QLabel(self.groupBox_LateralDose_crossline)
        self.label_LDBP_CL_MinMax_rval.setGeometry(QtCore.QRect(130, 220, 41, 21))

self.label_LDBP_CL_MinMax_rval.setObjectName(_fromUtf8("label_LDBP_CL_MinMa
x_rval"))
        self.label_LDBP_CL_MinMax_ival =
QtGui.QLabel(self.groupBox_LateralDose_crossline)
        self.label_LDBP_CL_MinMax_ival.setGeometry(QtCore.QRect(200, 220, 41, 21))
```

```
self.label_LDBP_CL_MinMax_ival.setObjectName(_fromUtf8("label_LDBP_CL_MinMa
x_ival"))
        self.label_LDBP_CL_Pen_high_rval =
QtGui.QLabel(self.groupBox_LateralDose_crossline)
        self.label_LDBP_CL_Pen_high_rval.setGeometry(QtCore.QRect(130, 140, 41,
21))

self.label_LDBP_CL_Pen_high_rval.setObjectName(_fromUtf8("label_LDBP_CL_Pen_
high_rval"))
        self.label_LDBP_CL_Pen_low_ival =
QtGui.QLabel(self.groupBox_LateralDose_crossline)
        self.label_LDBP_CL_Pen_low_ival.setGeometry(QtCore.QRect(200, 120, 41, 21))

self.label_LDBP_CL_Pen_low_ival.setObjectName(_fromUtf8("label_LDBP_CL_Pen_l
ow_ival"))
        self.label_LDBP_CL_Pen_low_rval =
QtGui.QLabel(self.groupBox_LateralDose_crossline)
        self.label_LDBP_CL_Pen_low_rval.setGeometry(QtCore.QRect(130, 120, 41, 21))

self.label_LDBP_CL_Pen_low_rval.setObjectName(_fromUtf8("label_LDBP_CL_Pen_l
ow_rval"))
        self.label_LDBP_CL_Pen_high =
QtGui.QLabel(self.groupBox_LateralDose_crossline)
        self.label_LDBP_CL_Pen_high.setGeometry(QtCore.QRect(20, 140, 91, 21))

self.label_LDBP_CL_Pen_high.setObjectName(_fromUtf8("label_LDBP_CL_Pen_high"
))
        self.label_LDBP_CL_Pen_low =
QtGui.QLabel(self.groupBox_LateralDose_crossline)
        self.label_LDBP_CL_Pen_low.setGeometry(QtCore.QRect(20, 120, 81, 21))

self.label_LDBP_CL_Pen_low.setObjectName(_fromUtf8("label_LDBP_CL_Pen_low"))
        self.label_LDBP_CL_Pen_high_ival =
QtGui.QLabel(self.groupBox_LateralDose_crossline)
        self.label_LDBP_CL_Pen_high_ival.setGeometry(QtCore.QRect(200, 140, 41,
21))

self.label_LDBP_CL_Pen_high_ival.setObjectName(_fromUtf8("label_LDBP_CL_Pen_
high_ival"))
        self.label_LDBP_CL_UnifReg_rval =
QtGui.QLabel(self.groupBox_LateralDose_crossline)
        self.label_LDBP_CL_UnifReg_rval.setGeometry(QtCore.QRect(130, 200, 71, 21))

self.label_LDBP_CL_UnifReg_rval.setObjectName(_fromUtf8("label_LDBP_CL_UnifRe
g_rval"))
        self.label_LDBP_CL_FieldSize_rval =
QtGui.QLabel(self.groupBox_LateralDose_crossline)
        self.label_LDBP_CL_FieldSize_rval.setGeometry(QtCore.QRect(130, 90, 41, 21))

self.label_LDBP_CL_FieldSize_rval.setObjectName(_fromUtf8("label_LDBP_CL_Field
Size_rval"))
        self.label_LDBP_CL_FieldSize_ival =
QtGui.QLabel(self.groupBox_LateralDose_crossline)
        self.label_LDBP_CL_FieldSize_ival.setGeometry(QtCore.QRect(200, 90, 41, 21))
```

```
self.label_LDBP_CL_FieldSize_ival.setObjectName(_fromUtf8("label_LDBP_CL_Field
Size_ival"))
        self.label_LDBP_CL_FieldSize =
QtGui.QLabel(self.groupBox_LateralDose_crossline)
        self.label_LDBP_CL_FieldSize.setGeometry(QtCore.QRect(20, 90, 46, 21))

self.label_LDBP_CL_FieldSize.setObjectName(_fromUtf8("label_LDBP_CL_FieldSize"
))
        self.label_LDBP_CL_MinMax =
QtGui.QLabel(self.groupBox_LateralDose_crossline)
        self.label_LDBP_CL_MinMax.setGeometry(QtCore.QRect(20, 220, 101, 21))

self.label_LDBP_CL_MinMax.setObjectName(_fromUtf8("label_LDBP_CL_MinMax"))
        self.label_LDBP_CL_LatOff =
QtGui.QLabel(self.groupBox_LateralDose_crossline)
        self.label_LDBP_CL_LatOff.setGeometry(QtCore.QRect(20, 170, 81, 21))
        self.label_LDBP_CL_LatOff.setObjectName(_fromUtf8("label_LDBP_CL_LatOff"))
        self.label_LDBP_CL_LatOff_ival =
QtGui.QLabel(self.groupBox_LateralDose_crossline)
        self.label_LDBP_CL_LatOff_ival.setGeometry(QtCore.QRect(200, 170, 41, 21))

self.label_LDBP_CL_LatOff_ival.setObjectName(_fromUtf8("label_LDBP_CL_LatOff_iv
al"))
        self.label_LDBP_CL_LatOff_tol =
QtGui.QLabel(self.groupBox_LateralDose_crossline)
        self.label_LDBP_CL_LatOff_tol.setGeometry(QtCore.QRect(280, 170, 41, 21))

self.label_LDBP_CL_LatOff_tol.setObjectName(_fromUtf8("label_LDBP_CL_LatOff_tol
"))
        self.label_LDBP_CL_Pen_high_tol =
QtGui.QLabel(self.groupBox_LateralDose_crossline)
        self.label_LDBP_CL_Pen_high_tol.setGeometry(QtCore.QRect(280, 140, 41, 21))

self.label_LDBP_CL_Pen_high_tol.setObjectName(_fromUtf8("label_LDBP_CL_Pen_h
igh_tol"))
        self.label_LDBP_CL_MinMax_tol =
QtGui.QLabel(self.groupBox_LateralDose_crossline)
        self.label_LDBP_CL_MinMax_tol.setGeometry(QtCore.QRect(280, 220, 41, 21))

self.label_LDBP_CL_MinMax_tol.setObjectName(_fromUtf8("label_LDBP_CL_MinMax
_tol"))
        self.label_LDBP_CL_Pen_low_tol =
QtGui.QLabel(self.groupBox_LateralDose_crossline)
        self.label_LDBP_CL_Pen_low_tol.setGeometry(QtCore.QRect(280, 120, 41, 21))

self.label_LDBP_CL_Pen_low_tol.setObjectName(_fromUtf8("label_LDBP_CL_Pen_lo
w_tol"))
        self.label_LDBP_CL_FieldSize_tol =
QtGui.QLabel(self.groupBox_LateralDose_crossline)
        self.label_LDBP_CL_FieldSize_tol.setGeometry(QtCore.QRect(280, 90, 41, 21))

self.label_LDBP_CL_FieldSize_tol.setObjectName(_fromUtf8("label_LDBP_CL_FieldSi
ze_tol"))
```

```
        self.label_LDBP_CL_tolerance =
QtGui.QLabel(self.groupBox_LateralDose_crossline)
        self.label_LDBP_CL_tolerance.setGeometry(QtCore.QRect(280, 60, 61, 21))
        font = QtGui.QFont()
        font.setBold(True)
        font.setWeight(75)
        self.label_LDBP_CL_tolerance.setFont(font)

self.label_LDBP_CL_tolerance.setObjectName(_fromUtf8("label_LDBP_CL_tolerance"
))
        self.label_LDBP_CL_realValue =
QtGui.QLabel(self.groupBox_LateralDose_crossline)
        self.label_LDBP_CL_realValue.setGeometry(QtCore.QRect(130, 60, 61, 21))
        font = QtGui.QFont()
        font.setBold(True)
        font.setWeight(75)
        self.label_LDBP_CL_realValue.setFont(font)

self.label_LDBP_CL_realValue.setObjectName(_fromUtf8("label_LDBP_CL_realValue"
))
        self.label_LDBP_CL_idealValue =
QtGui.QLabel(self.groupBox_LateralDose_crossline)
        self.label_LDBP_CL_idealValue.setGeometry(QtCore.QRect(200, 60, 71, 21))
        font = QtGui.QFont()
        font.setBold(True)
        font.setWeight(75)
        self.label_LDBP_CL_idealValue.setFont(font)

self.label_LDBP_CL_idealValue.setObjectName(_fromUtf8("label_LDBP_CL_idealValu
e"))
        self.label_LDBP_CL_yCoord =
QtGui.QLabel(self.groupBox_LateralDose_crossline)
        self.label_LDBP_CL_yCoord.setGeometry(QtCore.QRect(150, 20, 16, 21))

self.label_LDBP_CL_yCoord.setObjectName(_fromUtf8("label_LDBP_CL_yCoord"))
        self.label_LDBP_CL_zCoord =
QtGui.QLabel(self.groupBox_LateralDose_crossline)
        self.label_LDBP_CL_zCoord.setGeometry(QtCore.QRect(310, 20, 16, 21))

self.label_LDBP_CL_zCoord.setObjectName(_fromUtf8("label_LDBP_CL_zCoord"))
        self.label_LDBP_CL_yCoord_val =
QtGui.QLabel(self.groupBox_LateralDose_crossline)
        self.label_LDBP_CL_yCoord_val.setGeometry(QtCore.QRect(170, 20, 81, 21))

self.label_LDBP_CL_yCoord_val.setObjectName(_fromUtf8("label_LDBP_CL_yCoord_
val"))
        self.label_LDBP_CL_zCoord_val =
QtGui.QLabel(self.groupBox_LateralDose_crossline)
        self.label_LDBP_CL_zCoord_val.setGeometry(QtCore.QRect(330, 20, 81, 21))

self.label_LDBP_CL_zCoord_val.setObjectName(_fromUtf8("label_LDBP_CL_zCoord_
val"))
        self.label_LDBP_CL_xCoord_val =
QtGui.QLabel(self.groupBox_LateralDose_crossline)
        self.label_LDBP_CL_xCoord_val.setGeometry(QtCore.QRect(40, 20, 81, 21))
```

```
self.label_LDBP_CL_xCoord_val.setObjectName(_fromUtf8("label_LDBP_CL_xCoord_
val"))
        self.label_LDBP_CL_xCoord =
QtGui.QLabel(self.groupBox_LateralDose_crossline)
        self.label_LDBP_CL_xCoord.setGeometry(QtCore.QRect(20, 20, 16, 21))

self.label_LDBP_CL_xCoord.setObjectName(_fromUtf8("label_LDBP_CL_xCoord"))
        self.label_LDBP_CL_Deviation =
QtGui.QLabel(self.groupBox_LateralDose_crossline)
        self.label_LDBP_CL_Deviation.setGeometry(QtCore.QRect(350, 60, 61, 21))
        font = QtGui.QFont()
        font.setBold(True)
        font.setWeight(75)
        self.label_LDBP_CL_Deviation.setFont(font)

self.label_LDBP_CL_Deviation.setObjectName(_fromUtf8("label_LDBP_CL_Deviation"
))
        self.label_LDBP_CL_Pen_low_dev =
QtGui.QLabel(self.groupBox_LateralDose_crossline)
        self.label_LDBP_CL_Pen_low_dev.setGeometry(QtCore.QRect(350, 120, 41, 21))

self.label_LDBP_CL_Pen_low_dev.setObjectName(_fromUtf8("label_LDBP_CL_Pen_l
ow_dev"))
        self.label_LDBP_CL_LatOff_dev =
QtGui.QLabel(self.groupBox_LateralDose_crossline)
        self.label_LDBP_CL_LatOff_dev.setGeometry(QtCore.QRect(350, 170, 41, 21))

self.label_LDBP_CL_LatOff_dev.setObjectName(_fromUtf8("label_LDBP_CL_LatOff_d
ev"))
        self.label_LDBP_CL_MinMax_dev =
QtGui.QLabel(self.groupBox_LateralDose_crossline)
        self.label_LDBP_CL_MinMax_dev.setGeometry(QtCore.QRect(350, 220, 41, 21))

self.label_LDBP_CL_MinMax_dev.setObjectName(_fromUtf8("label_LDBP_CL_MinMa
x_dev"))
        self.label_LDBP_CL_FieldSize_dev =
QtGui.QLabel(self.groupBox_LateralDose_crossline)
        self.label_LDBP_CL_FieldSize_dev.setGeometry(QtCore.QRect(350, 90, 41, 21))

self.label_LDBP_CL_FieldSize_dev.setObjectName(_fromUtf8("label_LDBP_CL_Field
Size_dev"))
        self.label_LDBP_CL_Pen_high_dev =
QtGui.QLabel(self.groupBox_LateralDose_crossline)
        self.label_LDBP_CL_Pen_high_dev.setGeometry(QtCore.QRect(350, 140, 41,
21))

self.label_LDBP_CL_Pen_high_dev.setObjectName(_fromUtf8("label_LDBP_CL_Pen_
high_dev"))
        self.textEdit_MessageField = QtGui.QTextEdit(self.centralwidget)
        self.textEdit_MessageField.setGeometry(QtCore.QRect(20, 620, 301, 71))
        self.textEdit_MessageField.setReadOnly(True)
        self.textEdit_MessageField.setObjectName(_fromUtf8("textEdit_MessageField"))
        self.groupBox_DD_WedgePhantom = QtGui.QGroupBox(self.centralwidget)
```

```
        self.groupBox_DD_WedgePhantom.setGeometry(QtCore.QRect(10, 80, 421,
231))

self.groupBox_DD_WedgePhantom.setObjectName(_fromUtf8("groupBox_DD_Wedge
Phantom"))
        self.label_DDWP_R1 = QtGui.QLabel(self.groupBox_DD_WedgePhantom)
        self.label_DDWP_R1.setGeometry(QtCore.QRect(10, 80, 61, 21))
        self.label_DDWP_R1.setObjectName(_fromUtf8("label_DDWP_R1"))
        self.label_DDWP_R1_rval = QtGui.QLabel(self.groupBox_DD_WedgePhantom)
        self.label_DDWP_R1_rval.setGeometry(QtCore.QRect(60, 80, 41, 21))
        self.label_DDWP_R1_rval.setObjectName(_fromUtf8("label_DDWP_R1_rval"))
        self.label_DDWP_R1_ival = QtGui.QLabel(self.groupBox_DD_WedgePhantom)
        self.label_DDWP_R1_ival.setGeometry(QtCore.QRect(150, 80, 41, 21))
        self.label_DDWP_R1_ival.setObjectName(_fromUtf8("label_DDWP_R1_ival"))
        self.label_DDWP_R2 = QtGui.QLabel(self.groupBox_DD_WedgePhantom)
        self.label_DDWP_R2.setGeometry(QtCore.QRect(10, 110, 41, 21))
        self.label_DDWP_R2.setObjectName(_fromUtf8("label_DDWP_R2"))
        self.label_DDWP_P1 = QtGui.QLabel(self.groupBox_DD_WedgePhantom)
        self.label_DDWP_P1.setGeometry(QtCore.QRect(10, 140, 46, 21))
        self.label_DDWP_P1.setObjectName(_fromUtf8("label_DDWP_P1"))
        self.label_DDWP_P2 = QtGui.QLabel(self.groupBox_DD_WedgePhantom)
        self.label_DDWP_P2.setGeometry(QtCore.QRect(10, 160, 46, 21))
        self.label_DDWP_P2.setObjectName(_fromUtf8("label_DDWP_P2"))
        self.label_DDWP_P3 = QtGui.QLabel(self.groupBox_DD_WedgePhantom)
        self.label_DDWP_P3.setGeometry(QtCore.QRect(10, 180, 46, 21))
        self.label_DDWP_P3.setObjectName(_fromUtf8("label_DDWP_P3"))
        self.label_DDWP_P4 = QtGui.QLabel(self.groupBox_DD_WedgePhantom)
        self.label_DDWP_P4.setGeometry(QtCore.QRect(10, 200, 46, 21))
        self.label_DDWP_P4.setObjectName(_fromUtf8("label_DDWP_P4"))
        self.label_DDWP_R2_ival = QtGui.QLabel(self.groupBox_DD_WedgePhantom)
        self.label_DDWP_R2_ival.setGeometry(QtCore.QRect(150, 110, 41, 21))
        self.label_DDWP_R2_ival.setObjectName(_fromUtf8("label_DDWP_R2_ival"))
        self.label_DDWP_R2_rval = QtGui.QLabel(self.groupBox_DD_WedgePhantom)
        self.label_DDWP_R2_rval.setGeometry(QtCore.QRect(60, 110, 41, 21))
        self.label_DDWP_R2_rval.setObjectName(_fromUtf8("label_DDWP_R2_rval"))
        self.label_DDWP_P1_rval = QtGui.QLabel(self.groupBox_DD_WedgePhantom)
        self.label_DDWP_P1_rval.setGeometry(QtCore.QRect(60, 140, 81, 21))
        self.label_DDWP_P1_rval.setObjectName(_fromUtf8("label_DDWP_P1_rval"))
        self.label_DDWP_P1_ival = QtGui.QLabel(self.groupBox_DD_WedgePhantom)
        self.label_DDWP_P1_ival.setGeometry(QtCore.QRect(150, 140, 81, 21))
        self.label_DDWP_P1_ival.setObjectName(_fromUtf8("label_DDWP_P1_ival"))
        self.label_DDWP_P2_rval = QtGui.QLabel(self.groupBox_DD_WedgePhantom)
        self.label_DDWP_P2_rval.setGeometry(QtCore.QRect(60, 160, 81, 21))
        self.label_DDWP_P2_rval.setObjectName(_fromUtf8("label_DDWP_P2_rval"))
        self.label_DDWP_P2_ival = QtGui.QLabel(self.groupBox_DD_WedgePhantom)
        self.label_DDWP_P2_ival.setGeometry(QtCore.QRect(150, 160, 81, 21))
        self.label_DDWP_P2_ival.setObjectName(_fromUtf8("label_DDWP_P2_ival"))
        self.label_DDWP_P3_rval = QtGui.QLabel(self.groupBox_DD_WedgePhantom)
        self.label_DDWP_P3_rval.setGeometry(QtCore.QRect(60, 180, 81, 21))
        self.label_DDWP_P3_rval.setObjectName(_fromUtf8("label_DDWP_P3_rval"))
        self.label_DDWP_P3_ival = QtGui.QLabel(self.groupBox_DD_WedgePhantom)
        self.label_DDWP_P3_ival.setGeometry(QtCore.QRect(150, 180, 81, 21))
        self.label_DDWP_P3_ival.setObjectName(_fromUtf8("label_DDWP_P3_ival"))
        self.label_DDWP_P4_rval = QtGui.QLabel(self.groupBox_DD_WedgePhantom)
        self.label_DDWP_P4_rval.setGeometry(QtCore.QRect(60, 200, 81, 21))
```

```
        self.label_DDWP_P4_rval.setObjectName(_fromUtf8("label_DDWP_P4_rval"))
        self.label_DDWP_P4_ival = QtGui.QLabel(self.groupBox_DD_WedgePhantom)
        self.label_DDWP_P4_ival.setGeometry(QtCore.QRect(150, 200, 81, 21))
        self.label_DDWP_P4_ival.setObjectName(_fromUtf8("label_DDWP_P4_ival"))
        self.label_DDWP_RangeModulation =
QtGui.QLabel(self.groupBox_DD_WedgePhantom)
        self.label_DDWP_RangeModulation.setGeometry(QtCore.QRect(10, 20, 121, 21))

self.label_DDWP_RangeModulation.setObjectName(_fromUtf8("label_DDWP_RangeM
odulation"))
        self.label_DDWP_P1_tol = QtGui.QLabel(self.groupBox_DD_WedgePhantom)
        self.label_DDWP_P1_tol.setGeometry(QtCore.QRect(240, 140, 41, 21))
        self.label_DDWP_P1_tol.setObjectName(_fromUtf8("label_DDWP_P1_tol"))
        self.label_DDWP_R1_tol = QtGui.QLabel(self.groupBox_DD_WedgePhantom)
        self.label_DDWP_R1_tol.setGeometry(QtCore.QRect(240, 80, 41, 21))
        self.label_DDWP_R1_tol.setObjectName(_fromUtf8("label_DDWP_R1_tol"))
        self.label_DDWP_P2_tol = QtGui.QLabel(self.groupBox_DD_WedgePhantom)
        self.label_DDWP_P2_tol.setGeometry(QtCore.QRect(240, 160, 41, 21))
        self.label_DDWP_P2_tol.setObjectName(_fromUtf8("label_DDWP_P2_tol"))
        self.label_DDWP_P3_tol = QtGui.QLabel(self.groupBox_DD_WedgePhantom)
        self.label_DDWP_P3_tol.setGeometry(QtCore.QRect(240, 180, 41, 21))
        self.label_DDWP_P3_tol.setObjectName(_fromUtf8("label_DDWP_P3_tol"))
        self.label_DDWP_R2_tol = QtGui.QLabel(self.groupBox_DD_WedgePhantom)
        self.label_DDWP_R2_tol.setGeometry(QtCore.QRect(240, 110, 41, 21))
        self.label_DDWP_R2_tol.setObjectName(_fromUtf8("label_DDWP_R2_tol"))
        self.label_DDWP_P4_tol = QtGui.QLabel(self.groupBox_DD_WedgePhantom)
        self.label_DDWP_P4_tol.setGeometry(QtCore.QRect(240, 200, 41, 21))
        self.label_DDWP_P4_tol.setObjectName(_fromUtf8("label_DDWP_P4_tol"))
        self.label_DDWP_idealValue =
QtGui.QLabel(self.groupBox_DD_WedgePhantom)
        self.label_DDWP_idealValue.setGeometry(QtCore.QRect(150, 50, 71, 21))
        font = QtGui.QFont()
        font.setBold(True)
        font.setWeight(75)
        self.label_DDWP_idealValue.setFont(font)

self.label_DDWP_idealValue.setObjectName(_fromUtf8("label_DDWP_idealValue"))
        self.label_DDWP_realValue = QtGui.QLabel(self.groupBox_DD_WedgePhantom)
        self.label_DDWP_realValue.setGeometry(QtCore.QRect(60, 50, 61, 21))
        font = QtGui.QFont()
        font.setBold(True)
        font.setWeight(75)
        self.label_DDWP_realValue.setFont(font)

self.label_DDWP_realValue.setObjectName(_fromUtf8("label_DDWP_realValue"))
        self.label_DDWP_tolerance = QtGui.QLabel(self.groupBox_DD_WedgePhantom)
        self.label_DDWP_tolerance.setGeometry(QtCore.QRect(240, 50, 61, 21))
        font = QtGui.QFont()
        font.setBold(True)
        font.setWeight(75)
        self.label_DDWP_tolerance.setFont(font)
        self.label_DDWP_tolerance.setObjectName(_fromUtf8("label_DDWP_tolerance"))
        self.pushButton_DDWP_LoadOption =
QtGui.QPushButton(self.groupBox_DD_WedgePhantom)
```

```
        self.pushButton_DDWP_LoadOption.setGeometry(QtCore.QRect(250, 20, 75,
23))

self.pushButton_DDWP_LoadOption.setObjectName(_fromUtf8("pushButton_DDWP_L
oadOption"))
        self.lineEdit_DDWP_option =
QtGui.QLineEdit(self.groupBox_DD_WedgePhantom)
        self.lineEdit_DDWP_option.setGeometry(QtCore.QRect(210, 20, 31, 21))
        self.lineEdit_DDWP_option.setText(_fromUtf8(""))
        self.lineEdit_DDWP_option.setObjectName(_fromUtf8("lineEdit_DDWP_option"))
        self.label_DDWP_option = QtGui.QLabel(self.groupBox_DD_WedgePhantom)
        self.label_DDWP_option.setGeometry(QtCore.QRect(170, 20, 41, 21))
        self.label_DDWP_option.setObjectName(_fromUtf8("label_DDWP_option"))
        self.label_DDWP_Deviation = QtGui.QLabel(self.groupBox_DD_WedgePhantom)
        self.label_DDWP_Deviation.setGeometry(QtCore.QRect(320, 50, 61, 21))
        font = QtGui.QFont()
        font.setBold(True)
        font.setWeight(75)
        self.label_DDWP_Deviation.setFont(font)
        self.label_DDWP_Deviation.setObjectName(_fromUtf8("label_DDWP_Deviation"))
        self.label_DDWP_P4_dev = QtGui.QLabel(self.groupBox_DD_WedgePhantom)
        self.label_DDWP_P4_dev.setGeometry(QtCore.QRect(320, 200, 81, 21))
        self.label_DDWP_P4_dev.setObjectName(_fromUtf8("label_DDWP_P4_dev"))
        self.label_DDWP_P1_dev = QtGui.QLabel(self.groupBox_DD_WedgePhantom)
        self.label_DDWP_P1_dev.setGeometry(QtCore.QRect(320, 140, 81, 21))
        self.label_DDWP_P1_dev.setObjectName(_fromUtf8("label_DDWP_P1_dev"))
        self.label_DDWP_P2_dev = QtGui.QLabel(self.groupBox_DD_WedgePhantom)
        self.label_DDWP_P2_dev.setGeometry(QtCore.QRect(320, 160, 81, 21))
        self.label_DDWP_P2_dev.setObjectName(_fromUtf8("label_DDWP_P2_dev"))
        self.label_DDWP_R2_dev = QtGui.QLabel(self.groupBox_DD_WedgePhantom)
        self.label_DDWP_R2_dev.setGeometry(QtCore.QRect(320, 110, 81, 21))
        self.label_DDWP_R2_dev.setObjectName(_fromUtf8("label_DDWP_R2_dev"))
        self.label_DDWP_R1_dev = QtGui.QLabel(self.groupBox_DD_WedgePhantom)
        self.label_DDWP_R1_dev.setGeometry(QtCore.QRect(320, 80, 81, 21))
        self.label_DDWP_R1_dev.setObjectName(_fromUtf8("label_DDWP_R1_dev"))
        self.label_DDWP_P3_dev = QtGui.QLabel(self.groupBox_DD_WedgePhantom)
        self.label_DDWP_P3_dev.setGeometry(QtCore.QRect(320, 180, 81, 21))
        self.label_DDWP_P3_dev.setObjectName(_fromUtf8("label_DDWP_P3_dev"))
        self.widget_2 = MatplotLibWidget(self.centralwidget)
        self.widget_2.setGeometry(QtCore.QRect(440, 370, 501, 341))
        self.widget_2.setObjectName(_fromUtf8("widget_2"))
        self.widget_0 = MatplotLibWidget(self.centralwidget)
        self.widget_0.setGeometry(QtCore.QRect(440, 40, 501, 651))
        self.widget_0.setObjectName(_fromUtf8("widget_0"))
        self.labelAnalyseMenu = QtGui.QLabel(self.centralwidget)
        self.labelAnalyseMenu.setGeometry(QtCore.QRect(10, 0, 661, 31))
        font = QtGui.QFont()
        font.setFamily(_fromUtf8("MS Shell Dlg 2"))
        font.setPointSize(12)
        font.setBold(True)
        font.setWeight(75)
        self.labelAnalyseMenu.setFont(font)
        self.labelAnalyseMenu.setText(_fromUtf8(""))
        self.labelAnalyseMenu.setObjectName(_fromUtf8("labelAnalyseMenu"))
        self.label_image = QtGui.QLabel(self.centralwidget)
```

```
        self.label_image.setGeometry(QtCore.QRect(80, 340, 251, 251))
        self.label_image.setText(_fromUtf8(""))
        self.label_image.setObjectName(_fromUtf8("label_image"))
        self.pushButton_Export = QtGui.QPushButton(self.centralwidget)
        self.pushButton_Export.setGeometry(QtCore.QRect(310, 50, 121, 21))
        self.pushButton_Export.setObjectName(_fromUtf8("pushButton_Export"))
        self.label_master_fail_pass = QtGui.QLabel(self.centralwidget)
        self.label_master_fail_pass.setGeometry(QtCore.QRect(340, 620, 81, 61))
        font = QtGui.QFont()
        font.setPointSize(11)
        font.setBold(True)
        font.setWeight(75)
        self.label_master_fail_pass.setFont(font)
        self.label_master_fail_pass.setText(_fromUtf8(""))
        self.label_master_fail_pass.setAlignment(QtCore.Qt.AlignCenter)
        self.label_master_fail_pass.setObjectName(_fromUtf8("label_master_fail_pass"))
        MainWindow.setCentralWidget(self.centralwidget)
        self.menubar = QtGui.QMenuBar(MainWindow)
        self.menubar.setGeometry(QtCore.QRect(0, 0, 962, 21))
        self.menubar.setObjectName(_fromUtf8("menubar"))
        self.menuAnalysing = QtGui.QMenu(self.menubar)
        self.menuAnalysing.setObjectName(_fromUtf8("menuAnalysing"))
        self.menu_Daily_QA = QtGui.QMenu(self.menubar)
        self.menu_Daily_QA.setObjectName(_fromUtf8("menu_Daily_QA"))
        self.menuDS = QtGui.QMenu(self.menu_Daily_QA)
        self.menuDS.setObjectName(_fromUtf8("menuDS"))
        MainWindow.setMenuBar(self.menubar)
        self.statusbar = QtGui.QStatusBar(MainWindow)
        self.statusbar.setObjectName(_fromUtf8("statusbar"))
        MainWindow.setStatusBar(self.statusbar)
        self.actionDepth_Dose_WaterPhantom = QtGui.QAction(MainWindow)

self.actionDepth_Dose_WaterPhantom.setObjectName(_fromUtf8("actionDepth_Dose
_WaterPhantom"))
        self.actionLateral_Dose_WaterPhantom = QtGui.QAction(MainWindow)

self.actionLateral_Dose_WaterPhantom.setObjectName(_fromUtf8("actionLateral_Dos
e_WaterPhantom"))
        self.actionPBS = QtGui.QAction(MainWindow)
        self.actionPBS.setObjectName(_fromUtf8("actionPBS"))
        self.actionDepth_Dose_LYNX = QtGui.QAction(MainWindow)

self.actionDepth_Dose_LYNX.setObjectName(_fromUtf8("actionDepth_Dose_LYNX"))
        self.actionLateral_Profile_LYNX = QtGui.QAction(MainWindow)

self.actionLateral_Profile_LYNX.setObjectName(_fromUtf8("actionLateral_Profile_LYN
X"))
        self.actionExport_PDF = QtGui.QAction(MainWindow)
        self.actionExport_PDF.setObjectName(_fromUtf8("actionExport_PDF"))
        self.menuAnalysing.addAction(self.actionDepth_Dose_WaterPhantom)
        self.menuAnalysing.addAction(self.actionLateral_Dose_WaterPhantom)
        self.menuDS.addAction(self.actionDepth_Dose_LYNX)
        self.menuDS.addAction(self.actionLateral_Profile_LYNX)
        self.menu_Daily_QA.addAction(self.menuDS.menuAction())
        self.menubar.addAction(self.menu_Daily_QA.menuAction())
```

```
        self.menubar.addAction(self.menuAnalysing.menuAction())

        self.retranslateUi(MainWindow)
        QtCore.QMetaObject.connectSlotsByName(MainWindow)

    def retranslateUi(self, MainWindow):
        MainWindow.setWindowTitle(_translate("MainWindow", "AMODD - Analysing
Modul of Dose Distributions
Version 1.0", None))
        self.lineEdit_FilePath.setText(_translate("MainWindow", "ImportFilePath", None))
        self.groupBox_LateralDose_inline.setTitle(_translate("MainWindow", "Lateral
Dose - inline (Y)", None))
        self.label_LDBP_IL_FieldSize.setText(_translate("MainWindow", "Field Size",
None))
        self.label_LDBP_IL_Pen_low.setText(_translate("MainWindow", "Penumbra
x_low", None))
        self.label_LDBP_IL_Pen_high.setText(_translate("MainWindow", "Penumbra
x_high", None))
        self.label_LDBP_IL_LatOff.setText(_translate("MainWindow", "Lateral Offset",
None))
        self.label_LDBP_IL_UnifReg.setText(_translate("MainWindow", "Uniformity Re-
gion", None))
        self.label_LDBP_IL_MinMax.setText(_translate("MainWindow", "(Max - Min) /
Max", None))
        self.label_LDBP_IL_Pen_low_rval.setText(_translate("MainWindow", "wert",
None))
        self.label_LDBP_IL_UnifReg_rval.setText(_translate("MainWindow", "wert",
None))
        self.label_LDBP_IL_FieldSize_rval.setText(_translate("MainWindow", "wert",
None))
        self.label_LDBP_IL_LatOff_rval.setText(_translate("MainWindow", "wert", None))
        self.label_LDBP_IL_MinMax_rval.setText(_translate("MainWindow", "wert",
None))
        self.label_LDBP_IL_Pen_high_rval.setText(_translate("MainWindow", "wert",
None))
        self.label_LDBP_IL_Pen_high_ival.setText(_translate("MainWindow", "wert",
None))
        self.label_LDBP_IL_Pen_low_ival.setText(_translate("MainWindow", "wert",
None))
        self.label_LDBP_IL_MinMax_ival.setText(_translate("MainWindow", "wert",
None))
        self.label_LDBP_IL_FieldSize_ival.setText(_translate("MainWindow", "wert",
None))
        self.label_LDBP_IL_LatOff_ival.setText(_translate("MainWindow", "wert", None))
        self.label_LDBP_IL_Pen_low_tol.setText(_translate("MainWindow", "wert", None))
        self.label_LDBP_IL_FieldSize_tol.setText(_translate("MainWindow", "wert",
None))
        self.label_LDBP_IL_LatOff_tol.setText(_translate("MainWindow", "wert", None))
        self.label_LDBP_IL_Pen_high_tol.setText(_translate("MainWindow", "wert",
None))
        self.label_LDBP_IL_MinMax_tol.setText(_translate("MainWindow", "wert", None))
        self.label_LDBP_IL_relValue.setText(_translate("MainWindow", "Real Value",
None))
        self.label_LDBP_IL_tolerance.setText(_translate("MainWindow", "Tolerance",
None))
```

```
        self.label_LDBP_IL_idealValue.setText(_translate("MainWindow", "Reference",
None))
        self.label_LDBP_IL_xCoord.setText(_translate("MainWindow", "X:", None))
        self.label_LDBP_IL_yCoord.setText(_translate("MainWindow", "Y:", None))
        self.label_LDBP_IL_zCoord.setText(_translate("MainWindow", "Z:", None))
        self.label_LDBP_IL_xCoord_val.setText(_translate("MainWindow", "wert", None))
        self.label_LDBP_IL_yCoord_val.setText(_translate("MainWindow", "wert", None))
        self.label_LDBP_IL_zCoord_val.setText(_translate("MainWindow", "wert", None))
        self.label_LDBP_IL_Deviation.setText(_translate("MainWindow", "Deviation",
None))
        self.label_LDBP_IL_Pen_high_dev.setText(_translate("MainWindow", "wert",
None))
        self.label_LDBP_IL_FieldSize_dev.setText(_translate("MainWindow", "wert",
None))
        self.label_LDBP_IL_Pen_low_dev.setText(_translate("MainWindow", "wert",
None))
        self.label_LDBP_IL_MinMax_dev.setText(_translate("MainWindow", "wert",
None))
        self.label_LDBP_IL_LatOff_dev.setText(_translate("MainWindow", "wert", None))
        self.pushButton_LDBP_LoadOption.setText(_translate("MainWindow", "Load",
None))
        self.label_LDBP_option.setText(_translate("MainWindow", "RxxxMxxx:", None))
        self.label_LDBP_RangeModulation.setText(_translate("MainWindow", "Range &
Modulation", None))
        self.groupBox_DD_BluePhantom.setTitle(_translate("MainWindow", "Depth Dose
- Blue Phantom", None))
        self.label_DDBP_dR80_rval.setText(_translate("MainWindow", "wert", None))
        self.label_DDBP_dR90.setText(_translate("MainWindow", "dist R90[mm]", None))
        self.label_DDBP_dR20.setText(_translate("MainWindow", "dist R20[mm]", None))
        self.label_DDBP_dR90_rval.setText(_translate("MainWindow", "wert", None))
        self.label_DDBP_dR80.setText(_translate("MainWindow", "dist R80[mm]", None))
        self.label_DDBP_dR20_rval.setText(_translate("MainWindow", "wert", None))
        self.label_DDBP_SOBP_9090.setText(_translate("MainWindow", "SOBP(pR90-
dR90)[mm]", None))
        self.label_DDBP_DFO_rval.setText(_translate("MainWindow", "wert", None))
        self.label_DDBP_SOBP_9090_rval.setText(_translate("MainWindow", "wert",
None))
        self.label_DDBP_DFO.setText(_translate("MainWindow", "DFO(dR80-
dR20)[mm]", None))
        self.label_DDBP_dR80_ival.setText(_translate("MainWindow", "wert", None))
        self.label_DDBP_DFO_ival.setText(_translate("MainWindow", "wert", None))
        self.label_DDBP_dR90_ival.setText(_translate("MainWindow", "wert", None))
        self.label_DDBP_dR20_ival.setText(_translate("MainWindow", "wert", None))
        self.label_DDBP_SOBP_9090_ival.setText(_translate("MainWindow", "wert",
None))
        self.label_DDBP_pR98_rval.setText(_translate("MainWindow", "wert", None))
        self.label_DDBP_pR95_ival.setText(_translate("MainWindow", "wert", None))
        self.label_DDBP_pR90_ival.setText(_translate("MainWindow", "wert", None))
        self.label_DDBP_pR98.setText(_translate("MainWindow", "prox R98[mm]",
None))
        self.label_DDBP_pR95.setText(_translate("MainWindow", "prox R95[mm]",
None))
        self.label_DDBP_pR90.setText(_translate("MainWindow", "prox R90[mm]",
None))
        self.label_DDBP_pR98_ival.setText(_translate("MainWindow", "wert", None))
```

```
        self.label_DDBP_pR95_rval.setText(_translate("MainWindow", "wert", None))
        self.label_DDBP_pR90_rval.setText(_translate("MainWindow", "wert", None))
        self.label_DDBP_dR10.setText(_translate("MainWindow", "dist R10[mm]", None))
        self.label_DDBP_dR10_ival.setText(_translate("MainWindow", "wert", None))
        self.label_DDBP_dR10_rval.setText(_translate("MainWindow", "wert", None))
        self.label_DDBP_dR50.setText(_translate("MainWindow", "dist R50[mm]", None))
        self.label_DDBP_dR50_ival.setText(_translate("MainWindow", "wert", None))
        self.label_DDBP_dR50_rval.setText(_translate("MainWindow", "wert", None))
        self.label_DDBP_dR95.setText(_translate("MainWindow", "dist R95[mm]", None))
        self.label_DDBP_dR95_ival.setText(_translate("MainWindow", "wert", None))
        self.label_DDBP_dR95_rval.setText(_translate("MainWindow", "wert", None))
        self.label_DDBP_SOBP_9890_rval.setText(_translate("MainWindow", "wert",
None))
        self.label_DDBP_SOBP_9890.setText(_translate("MainWindow", "SOBP(pR98-
dR90)[mm]", None))
        self.label_DDBP_SOBP_9890_ival.setText(_translate("MainWindow", "wert",
None))
        self.label_DDBP_MinMax_ival.setText(_translate("MainWindow", "wert", None))
        self.label_DDBP_MinMax.setText(_translate("MainWindow", "(Max - Min) / Max",
None))
        self.label_DDBP_MinMax_rval.setText(_translate("MainWindow", "wert", None))
        self.label_DDBP_MinMax_tol.setText(_translate("MainWindow", "wert", None))
        self.label_DDBP_SOBP_9090_tol.setText(_translate("MainWindow", "wert",
None))
        self.label_DDBP_pR95_tol.setText(_translate("MainWindow", "wert", None))
        self.label_DDBP_DFO_tol.setText(_translate("MainWindow", "wert", None))
        self.label_DDBP_dR95_tol.setText(_translate("MainWindow", "wert", None))
        self.label_DDBP_dR20_tol.setText(_translate("MainWindow", "wert", None))
        self.label_DDBP_SOBP_9890_tol.setText(_translate("MainWindow", "wert",
None))
        self.label_DDBP_dR50_tol.setText(_translate("MainWindow", "wert", None))
        self.label_DDBP_pR90_tol.setText(_translate("MainWindow", "wert", None))
        self.label_DDBP_pR98_tol.setText(_translate("MainWindow", "wert", None))
        self.label_DDBP_dR80_tol.setText(_translate("MainWindow", "wert", None))
        self.label_DDBP_dR90_tol.setText(_translate("MainWindow", "wert", None))
        self.label_DDBP_dR10_tol.setText(_translate("MainWindow", "wert", None))
        self.label_DDBP_realValue.setText(_translate("MainWindow", "Real Value",
None))
        self.label_DDBP_idealValue.setText(_translate("MainWindow", "Reference",
None))
        self.label_DDBP_tolerance.setText(_translate("MainWindow", "Tolerance", None))
        self.label_DDBP_RangeModulation.setText(_translate("MainWindow", "Range &
Modulation", None))
        self.label_DDBP_option.setText(_translate("MainWindow", "RxxxMxxx:", None))
        self.pushButton_DDBP_LoadOption.setText(_translate("MainWindow", "Load",
None))
        self.label_DDBP_Deviation.setText(_translate("MainWindow", "Deviation", None))
        self.label_DDBP_pR98_dev.setText(_translate("MainWindow", "wert", None))
        self.label_DDBP_pR95_dev.setText(_translate("MainWindow", "wert", None))
        self.label_DDBP_dR90_dev.setText(_translate("MainWindow", "wert", None))
        self.label_DDBP_dR95_dev.setText(_translate("MainWindow", "wert", None))
        self.label_DDBP_SOBP_9890_dev.setText(_translate("MainWindow", "wert",
None))
        self.label_DDBP_pR90_dev.setText(_translate("MainWindow", "wert", None))
```

```
        self.label_DDBP_SOBP_9090_dev.setText(_translate("MainWindow", "wert",
None))
        self.label_DDBP_dR80_dev.setText(_translate("MainWindow", "wert", None))
        self.label_DDBP_DFO_dev.setText(_translate("MainWindow", "wert", None))
        self.label_DDBP_dR20_dev.setText(_translate("MainWindow", "wert", None))
        self.label_DDBP_dR10_dev.setText(_translate("MainWindow", "wert", None))
        self.label_DDBP_MinMax_dev.setText(_translate("MainWindow", "wert", None))
        self.label_DDBP_dR50_dev.setText(_translate("MainWindow", "wert", None))
        self.groupBox_LateralDose_crossline.setTitle(_translate("MainWindow", "Lateral
Dose - crossline (X)", None))
        self.label_LDBP_CL_LatOff_rval.setText(_translate("MainWindow", "wert", None))
        self.label_LDBP_CL_UnifReg.setText(_translate("MainWindow", "Uniformity Re-
gion", None))
        self.label_LDBP_CL_MinMax_rval.setText(_translate("MainWindow", "wert",
None))
        self.label_LDBP_CL_MinMax_ival.setText(_translate("MainWindow", "wert",
None))
        self.label_LDBP_CL_Pen_high_rval.setText(_translate("MainWindow", "wert",
None))
        self.label_LDBP_CL_Pen_low_ival.setText(_translate("MainWindow", "wert",
None))
        self.label_LDBP_CL_Pen_low_rval.setText(_translate("MainWindow", "wert",
None))
        self.label_LDBP_CL_Pen_high.setText(_translate("MainWindow", "Penumbra
y_high", None))
        self.label_LDBP_CL_Pen_low.setText(_translate("MainWindow", "Penumbra
y_low", None))
        self.label_LDBP_CL_Pen_high_ival.setText(_translate("MainWindow", "wert",
None))
        self.label_LDBP_CL_UnifReg_rval.setText(_translate("MainWindow", "wert",
None))
        self.label_LDBP_CL_FieldSize_rval.setText(_translate("MainWindow", "wert",
None))
        self.label_LDBP_CL_FieldSize_ival.setText(_translate("MainWindow", "wert",
None))
        self.label_LDBP_CL_FieldSize.setText(_translate("MainWindow", "Field Size",
None))
        self.label_LDBP_CL_MinMax.setText(_translate("MainWindow", "(Max - Min) /
Max", None))
        self.label_LDBP_CL_LatOff.setText(_translate("MainWindow", "Lateral Offset",
None))
        self.label_LDBP_CL_LatOff_ival.setText(_translate("MainWindow", "wert", None))
        self.label_LDBP_CL_LatOff_tol.setText(_translate("MainWindow", "wert", None))
        self.label_LDBP_CL_Pen_high_tol.setText(_translate("MainWindow", "wert",
None))
        self.label_LDBP_CL_MinMax_tol.setText(_translate("MainWindow", "wert",
None))
        self.label_LDBP_CL_Pen_low_tol.setText(_translate("MainWindow", "wert",
None))
        self.label_LDBP_CL_FieldSize_tol.setText(_translate("MainWindow", "wert",
None))
        self.label_LDBP_CL_tolerance.setText(_translate("MainWindow", "Tolerance",
None))
        self.label_LDBP_CL_realValue.setText(_translate("MainWindow", "Real Value",
None))
```

```
        self.label_LDBP_CL_idealValue.setText(_translate("MainWindow", "Reference",
None))
        self.label_LDBP_CL_yCoord.setText(_translate("MainWindow", "Y:", None))
        self.label_LDBP_CL_zCoord.setText(_translate("MainWindow", "Z:", None))
        self.label_LDBP_CL_yCoord_val.setText(_translate("MainWindow", "wert", None))
        self.label_LDBP_CL_zCoord_val.setText(_translate("MainWindow", "wert", None))
        self.label_LDBP_CL_xCoord_val.setText(_translate("MainWindow", "wert", None))
        self.label_LDBP_CL_xCoord.setText(_translate("MainWindow", "X:", None))
        self.label_LDBP_CL_Deviation.setText(_translate("MainWindow", "Deviation",
None))
        self.label_LDBP_CL_Pen_low_dev.setText(_translate("MainWindow", "wert",
None))
        self.label_LDBP_CL_LatOff_dev.setText(_translate("MainWindow", "wert", None))
        self.label_LDBP_CL_MinMax_dev.setText(_translate("MainWindow", "wert",
None))
        self.label_LDBP_CL_FieldSize_dev.setText(_translate("MainWindow", "wert",
None))
        self.label_LDBP_CL_Pen_high_dev.setText(_translate("MainWindow", "wert",
None))
        self.groupBox_DD_WedgePhantom.setTitle(_translate("MainWindow", "Depth
Dose - Wedge Phantom - LYNX", None))
        self.label_DDWP_R1.setText(_translate("MainWindow", "R1 [mm]", None))
        self.label_DDWP_R1_rval.setText(_translate("MainWindow", "wert", None))
        self.label_DDWP_R1_ival.setText(_translate("MainWindow", "wert", None))
        self.label_DDWP_R2.setText(_translate("MainWindow", "R2 [mm]", None))
        self.label_DDWP_P1.setText(_translate("MainWindow", "P1", None))
        self.label_DDWP_P2.setText(_translate("MainWindow", "P2", None))
        self.label_DDWP_P3.setText(_translate("MainWindow", "P3", None))
        self.label_DDWP_P4.setText(_translate("MainWindow", "P4", None))
        self.label_DDWP_R2_ival.setText(_translate("MainWindow", "wert", None))
        self.label_DDWP_R2_rval.setText(_translate("MainWindow", "wert", None))
        self.label_DDWP_P1_rval.setText(_translate("MainWindow", "wert", None))
        self.label_DDWP_P1_ival.setText(_translate("MainWindow", "wert", None))
        self.label_DDWP_P2_rval.setText(_translate("MainWindow", "wert", None))
        self.label_DDWP_P2_ival.setText(_translate("MainWindow", "wert", None))
        self.label_DDWP_P3_rval.setText(_translate("MainWindow", "wert", None))
        self.label_DDWP_P3_ival.setText(_translate("MainWindow", "wert", None))
        self.label_DDWP_P4_rval.setText(_translate("MainWindow", "wert", None))
        self.label_DDWP_P4_ival.setText(_translate("MainWindow", "wert", None))
        self.label_DDWP_RangeModulation.setText(_translate("MainWindow", "Range &
Modulation", None))
        self.label_DDWP_P1_tol.setText(_translate("MainWindow", "wert", None))
        self.label_DDWP_R1_tol.setText(_translate("MainWindow", "wert", None))
        self.label_DDWP_P2_tol.setText(_translate("MainWindow", "wert", None))
        self.label_DDWP_P3_tol.setText(_translate("MainWindow", "wert", None))
        self.label_DDWP_R2_tol.setText(_translate("MainWindow", "wert", None))
        self.label_DDWP_P4_tol.setText(_translate("MainWindow", "wert", None))
        self.label_DDWP_idealValue.setText(_translate("MainWindow", "Reference",
None))
        self.label_DDWP_realValue.setText(_translate("MainWindow", "Real Value",
None))
        self.label_DDWP_tolerance.setText(_translate("MainWindow", "Tolerance",
None))
        self.pushButton_DDWP_LoadOption.setText(_translate("MainWindow", "Load",
None))
```

```
        self.label_DDWP_option.setText(_translate("MainWindow", "Option:", None))
        self.label_DDWP_Deviation.setText(_translate("MainWindow", "Deviation", None))
        self.label_DDWP_P4_dev.setText(_translate("MainWindow", "wert", None))
        self.label_DDWP_P1_dev.setText(_translate("MainWindow", "wert", None))
        self.label_DDWP_P2_dev.setText(_translate("MainWindow", "wert", None))
        self.label_DDWP_R2_dev.setText(_translate("MainWindow", "wert", None))
        self.label_DDWP_R1_dev.setText(_translate("MainWindow", "wert", None))
        self.label_DDWP_P3_dev.setText(_translate("MainWindow", "wert", None))
        self.pushButton_Export.setText(_translate("MainWindow", "Export", None))
        self.menuAnalysing.setTitle(_translate("MainWindow", "Water Phantom", None))
        self.menu_Daily_QA.setTitle(_translate("MainWindow", "Daily QA", None))
        self.menuDS.setTitle(_translate("MainWindow", "DS", None))
        self.actionDepth_Dose_WaterPhantom.setText(_translate("MainWindow", "Depth
Dose", None))
        self.actionLateral_Dose_WaterPhantom.setText(_translate("MainWindow", "Lat-
eral Dose", None))
        self.actionPBS.setText(_translate("MainWindow", "PBS", None))
        self.actionDepth_Dose_LYNX.setText(_translate("MainWindow", "Depth Dose -
LYNX", None))
        self.actionLateral_Profile_LYNX.setText(_translate("MainWindow", "Lateral Profile
- LYNX", None))
        self.actionExport_PDF.setText(_translate("MainWindow", "Export PDF", None))

from matplotlibwidgetFile import MatplotLibWidget
```

Anhang A.3 – matplotlibwidgetFile

```
"""
Given a QWidget object these objects give a matplotlib window in Qt4
with one subplot and the toolbar. Can be used in designer-qt4 for promoting
QWidgets.
"""

from PyQt4 import QtGui
from matplotlib.backends.backend_qt4agg import FigureCanvasQTAgg as FigureCan-
vas
from matplotlib.backends.backend_qt4 import NavigationToolbar2QT as Navigation-
Toolbar

from matplotlib.figure import Figure


class MplCanvas(FigureCanvas):
    """
    Defines the canvas of the matplotlib window
    """
    def __init__(self):
        self.fig = Figure()                     # create figure
        self.axes = self.fig.add_subplot(111)       # create subplot
        self.fig.subplots_adjust(left=0.12, bottom=0.15, right=0.97,
                                 top=0.85,  wspace=None, hspace=None)

        FigureCanvas.__init__(self, self.fig)       # initialize canvas
        FigureCanvas.setSizePolicy(self, QtGui.QSizePolicy.Expanding,
                                   QtGui.QSizePolicy.Expanding)
        FigureCanvas.updateGeometry(self)


class MatplotLibWidget(QtGui.QWidget):
    """
    The matplotlibWidget class based on QWidget
    """
    def __init__(self, parent=None):
        QtGui.QWidget.__init__(self, parent)
        # save canvas and toolbar
        self.canvas = MplCanvas()
        self.toolbar = NavigationToolbar(self.canvas, self)
        # set layout and add them to widget
        self.vbl = QtGui.QVBoxLayout()
        self.vbl.addWidget(self.toolbar)
        self.vbl.addWidget(self.canvas)
        self.setLayout(self.vbl)

    def savePlot(self,filePath):
            self.canvas.fig.savefig(filePath)
```

Anhang B - Beispiel eines Exports eines Textfiles mit den berechneten Werten

Quality Assurance

Double Scattering
Depth Dose

LYNX 2D
Wedged Phantom

File name : 4_B4a_11s.ilx
Date of file : 08/05/2014
Time of file : 16:58:36
Date of interpretation : 7/7/2014
Time of interpretation : 21:39:32
Notes : B4a, 11s

Option : B4a
Range : 10.5 cm
Modulation : 5.5 cm

	Measured Value	Reference Value	Tolerance Value	Deviation Value
R1[mm]:	160.1	100.0	10.0	60.1
R2[mm]:	160.3	100.0	10.0	60.3
P1:	82.8/85.3/88.2	70.0/50.0/30.0	15.0	12.8/35.3/58.2
P2:	- - -	-		
P3:	82.8/85.9/90.0	70.0/50.0/30.0	15.0	12.8/35.9/60.0
P4:	- - -	-		

Anhang C - Beispiel-Report Auswertung Tiefendosisverteilung am Keilphantom

Quality Assurance

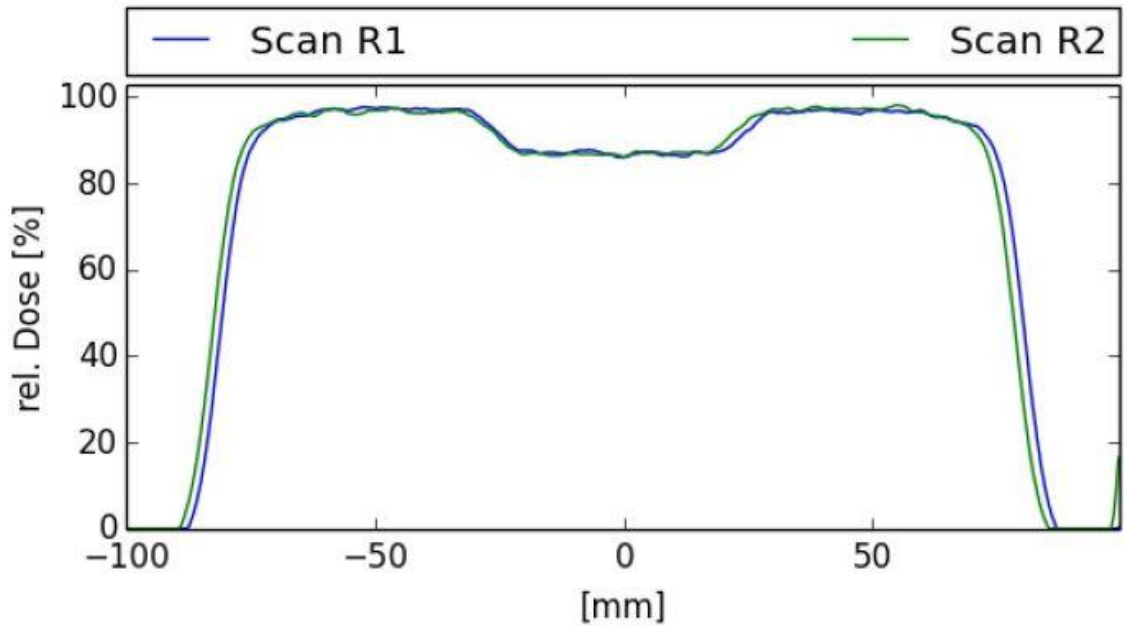
Double Scattering
Depth Dose

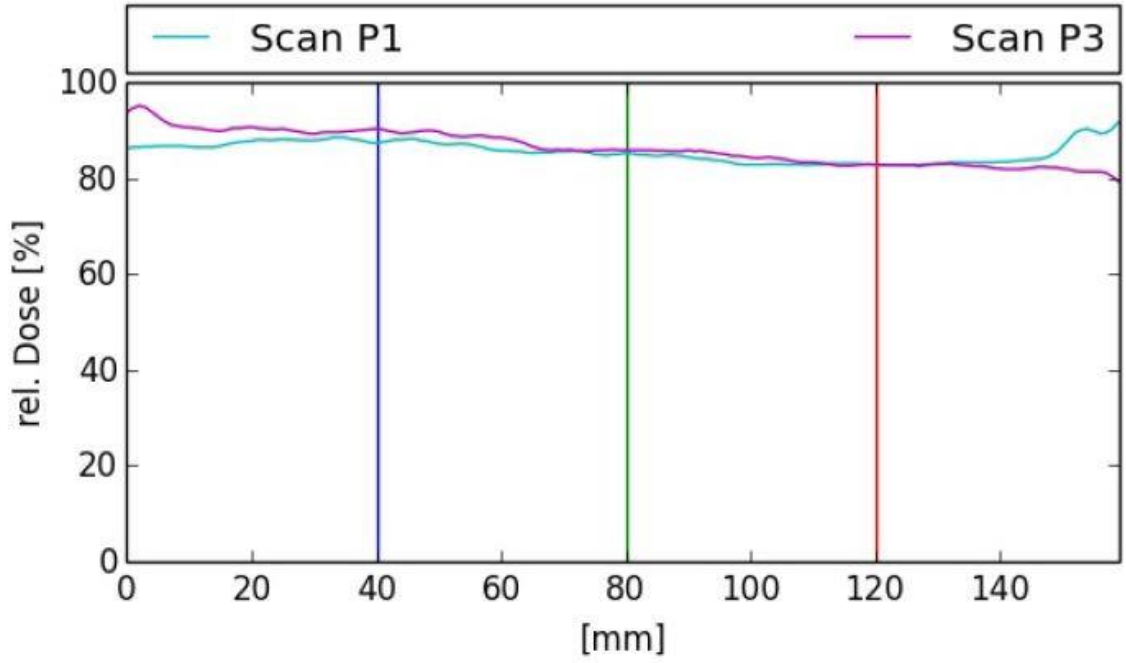
LYNX 2D
Wedged Phantom

File name : 4_B4a_11s.ilx
Date of file : 08/05/2014
Time of file : 16:58:36
Date of interpretation : 7/7/2014
Time of interpretation : 21:39:32
Notes : B4a, 11s

Option : B4a
Range : 10.5 cm
Modulation : 5.5 cm

	Measured Value	Reference Value	Tolerance Value	Deviation Value
R1[mm]:	160.1	100.0	10.0	60.1
R2[mm]:	160.3	100.0	10.0	60.3
P1:	82.8/85.3/88.2	70.0/50.0/30.0	15.0	12.8/35.3/58.2
P2:	- - -	-		
P3:	82.8/85.9/90.0	70.0/50.0/30.0	15.0	12.8/35.9/60.0
P4:	- - -	-		


Scan R1
Scan R2
rel. Dose [%]
100
80
60
40
20
0
−100
−50
0
50
[mm]


Scan P1
Scan P3
rel. Dose [%]
100
80
60
40
20
0
0
20
40
60
80
100
120
140
[mm]